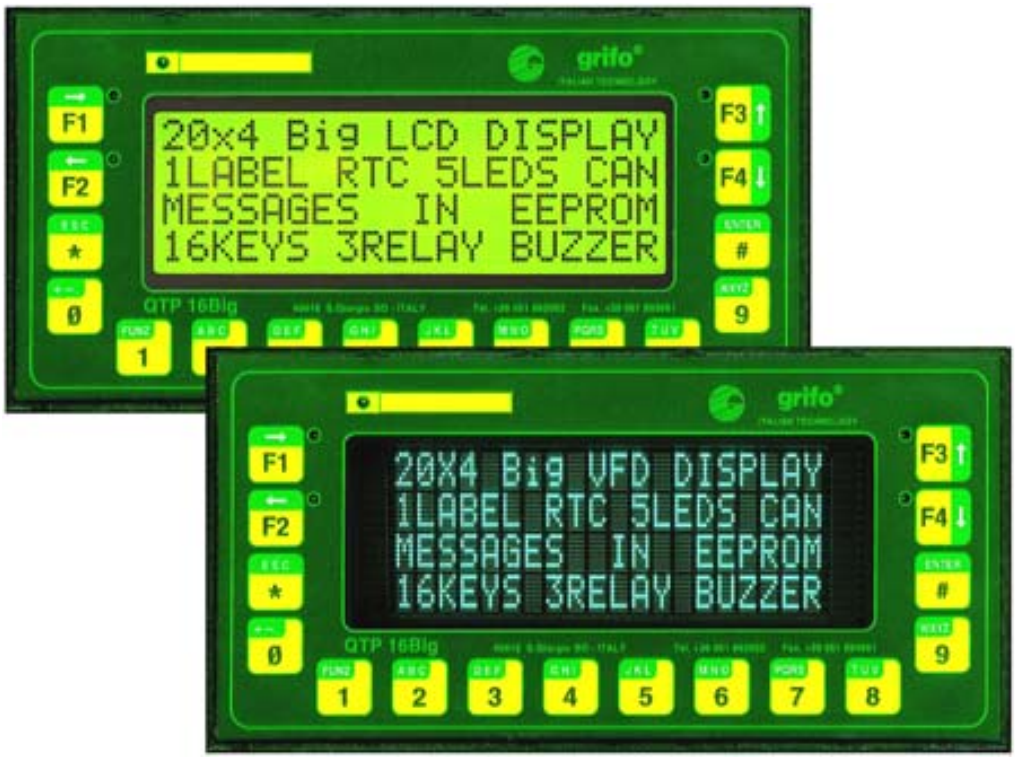
Libreria QTP 16Big

Libreria per Quick Terminal Panel 16 tasti, display grande

MANUALE UTENTE

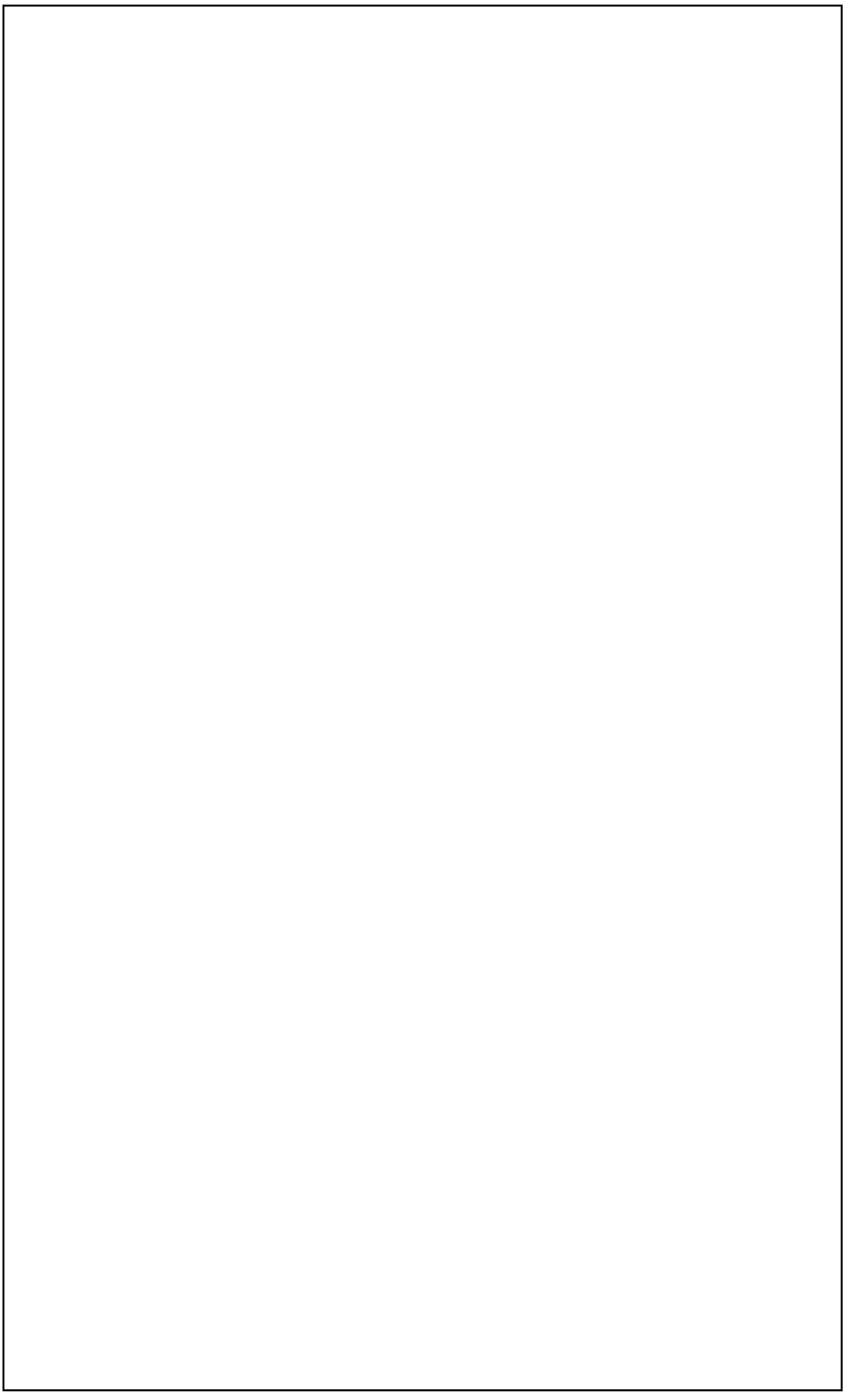


grifo®
ITALIAN TECHNOLOGY

Via dell' Artigiano, 8/6
40016 San Giorgio di Piano
(Bologna) ITALY
E-mail: grifo@grifo.it
http://www.grifo.it http://www.grifo.com
Tel. +39 051 892.052 (r.a.) FAX: +39 051 893.661

QTP 16Big.LIB Rel. 3.00 *Edizione 06 Ottobre 2006*

abaco, **GPC®**, **grifo®**, *sono marchi registrati della ditta* **grifo®**



Libreria QTP 16Big

Libreria per Quick Terminal Panel 16 tasti, display grande

MANUALE UTENTE

La libreria **QTP 16Big** coincide con un firmware appositamente realizzato per consentire all'utente di sviluppare un proprio **Programma Applicativo** in modo comodo, veloce ed efficace. Con questa libreria la **QTP 16Big** opera come un potente **Controllore Completo di Interfaccia Operatore** che può quindi funzionare sia autonomamente che abbinato ad altri sistemi.

Una ricca e completa serie di **Comandi**, che possono essere chiamati direttamente con i relativi **Parametri** e **Risultati**, consentono ad ogni utente di realizzare il programma applicativo che soddisfi al meglio le sue esigenze, con un **Tempo di Sviluppo** veramente **Ridotto**. Tali comandi soddisfano le normali richieste dell'ambiente industriale e sono il frutto di una esperienza decennale in questo settore.

La **QTP 16Big.LIB** non é quindi un prodotto finito pronto per essere installato ma deve essere prima **Specializzato** dall'utente. Questa specializzazione può essere effettuata con comodi ed economici **Ambienti di Sviluppo**, sia ad alto che a basso livello, e rendono la **QTP** un prodotto veramente flessibile e versatile. Infatti il programma applicativo dell'utente che la specializza consente di risolvere ogni problematica anche di alta complessità, ed allo stesso tempo consente di realizzare automazioni diverse, usando lo stesso hardware.

La maggioranza degli ambienti di sviluppo disponibili per la famiglia **I51** possono usare la libreria **QTP 16Big** e mettono a disposizione comode modalità di **Debug** del programma applicativo, riducendo ancora i tempi complessivi di preparazione. Tra questi si possono ricordare: **Assemblatori**; compilatori **PASCAL** (SYS51PW); compilatori **C** (HTC-51, SYS51CW, µC/51); programmazione a **Contatti** (LadderWORK); compilatori **BASIC** (BASCOM-8051); ecc.

grifo®
ITALIAN TECHNOLOGY
Via dell' Artigiano, 8/6
40016 San Giorgio di Piano
(Bologna) ITALY
E-mail: grifo@grifo.it
http://www.grifo.it http://www.grifo.com
Tel. +39 051 892.052 (r.a.) FAX: +39 051 893.661

QTP 16Big.LIB Rel. 3.00 *Edizione 06 Ottobre 2006*

abaco®, **GPC®**, grifo®, *sono marchi registrati della ditta* grifo®

Vincoli sulla documentazione grifo® Tutti i Diritti Riservati

Nessuna parte del presente manuale può essere riprodotta, trasmessa, trascritta, memorizzata in un archivio o tradotta in altre lingue, con qualunque forma o mezzo, sia esso elettronico, meccanico, magnetico ottico, chimico, manuale, senza il permesso scritto della **grifo®**.

IMPORTANTE

Tutte le informazioni contenute sul presente manuale sono state accuratamente verificate, ciononostante **grifo®** non si assume nessuna responsabilità per danni, diretti o indiretti, a cose e/o persone derivanti da errori, omissioni o dall'uso del presente manuale, del software o dell' hardware ad esso associato.
grifo® altresi si riserva il diritto di modificare il contenuto e la veste di questo manuale senza alcun preavviso, con l' intento di offrire un prodotto sempre migliore, senza che questo rappresenti un obbligo per **grifo®**.
Per le informazioni specifiche dei componenti utilizzati sui nostri prodotti, l'utente deve fare riferimento agli specifici Data Book delle case costruttrici o delle seconde sorgenti.

LEGENDA SIMBOLI

Nel presente manuale possono comparire i seguenti simboli:

Attenzione: Pericolo generico

Attenzione: Pericolo di alta tensione

Attenzione: Dispositivo sensibile alle cariche elettrostatiche

MARCHI REGISTRATI

abaco, **GPC®**, **grifo®** : sono marchi registrati della **grifo®**.
Altre marche o nomi di prodotti sono marchi registrati dei rispettivi proprietari.

INDICE GENERALE

INTRODUZIONE 1

VERSIONE 3

INDICAZIONI GENERALI 4

MATERIALE NECESSARIO 6
QTP 16BIG 6
PERSONAL COMPUTER 6
CAVO DI COMUNICAZIONE SERIALE 6
SOFTWARE DI LAVORO 7
PROGRAMMA SCRITTURA FLASH EPROM: FLIP 7
PROGRAMMA EMULAZIONE TERMINALE SERIALE 7
AMBIENTE DI SVILUPPO PROGRAMMA APPLICATIVO 8

DESCRIZIONE LIBRERIA QTP 16BIG 9
CONFIGURAZIONE QTP 16BIG 9
BUFFER DI COMUNICAZIONE 9
INTEGRAZIONE ED USO DELLA LIBRERIA 10
RISORSE USATE DALLA LIBRERIA 14
PROGRAMMAZIONE FLASH EPROM 16
SETUP LOCALE 19
MODALITÀ DI COMUNICAZIONE 20
PROGRAMMI DEMO PER FIRMWARE DI LIBRERIA 20

UTILIZZO LIBRERIA QTP 16BIG CON µC/51 22
INTEGRAZIONE LIBRERIA NEL µC/51 22
FILES FORNITI CON µC/51 23
COME INIZARE CON LIBRERIA QTP 16BIG E µC/51 24

UTILIZZO LIBRERIA QTP 16BIG CON BASCOM 8051 28
INTEGRAZIONE LIBRERIA NEL BASCOM 8051 28
FILES FORNITI CON BASCOM 8051 29
COME INIZARE CON LIBRERIA QTP 16BIG E BASCOM 8051 30

BIBLIOGRAFIA 35

APPENDICE A: DESCRIZIONE COMPONENTI DI BORDO A-1
MICROCONTROLLORE T89C5115 O T89C51CC02 ..... A-1
FAMIGLIA I51 ..... A-2
EEPROM OPZIONALE A-3
SRAM+RTC PCF8583 ..... A-4

APPENDICE B: INDICE ANALITICO B-1

INDICE DELLE FIGURE

FIGURA 1: ESEMPIO DI APPLICAZIONE CON QTP 16BIG.LIB 5
FIGURA 2: COLLEGAMENTO SERIALE TRA QTP 16BIG E PC DI SVILUPPO 6
FIGURA 3: ORGANIZZAZIONE AREA CODICE CON LIBRERIA 11
FIGURA 4: USO RAM CON LIBRERIA 15
FIGURA 5: FINESTRA SETTAGGIO FLIP (1 DI 4) 16
FIGURA 6: FINESTRA SETTAGGIO FLIP (2 DI 4) 17
FIGURA 7: FINESTRA SETTAGGIO FLIP (3 DI 4) 17
FIGURA 8: FINESTRA SETTAGGIO FLIP (4 DI 4) 18
FIGURA 9: MODALITÀ SVILUPPO CON LIBRERIA 19
FIGURA 10: SCHEMA DI COMUNICAZIONE SERIALE 21
FIGURA 11: ESECUZIONE PROGRAMMA DEMO IN µC/51 25
FIGURA 12: PROGRAMMAZIONE FLASH EPROM CON µC/51 26
FIGURA 13: COMPILAZIONE PROGRAMMA CON µC/51 27
FIGURA 14: CONFIGURAZIONE SERIALE CON BASCOM 8051 30
FIGURA 15: ESECUZIONE PROGRAMMA DEMO IN BASCOM 8051 31
FIGURA 16: CONFIGURAZIONE PROGRAMMATORE CON BASCOM 8051 32
FIGURA 17: PROGRAMMAZIONE FLASH EPROM CON BASCOM 8051 32
FIGURA 18: CONFIGURAZIONE COMPILATORE CON BASCOM 8051 33
FIGURA 19: COMPILAZIONE PROGRAMMA CON BASCOM 8051 34

INTRODUZIONE

L'uso di questi dispositivi è rivolto - **IN VIA ESCLUSIVA** - a personale specializzato.

Questo prodotto non è un **componente di sicurezza** così come definito dalla direttiva **98-37/CE**.

I pin della/e scheda/e non sono dotati di protezione contro le cariche elettrostatiche. Visto che esiste un collegamento diretto tra numerosi pin delle schede ed i rispettivi pin dei componenti di bordo, e che quest'ultimi sono sensibili ai fenomeni ESD, il personale che maneggia la/e scheda/e è invitato a prendere tutte le precauzioni necessarie per evitare i possibili danni che potrebbero derivare dalle cariche elettorostatiche.

Scopo di questo manuale è la trasmissione delle informazioni necessarie all'uso competente e sicuro dei prodotti. Esse sono il frutto di un'elaborazione continua e sistematica di dati e prove tecniche registrate e validate dal Costruttore, in attuazione alle procedure interne di sicurezza e qualità dell'informazione.

I dati di seguito riportati sono destinati - **IN VIA ESCLUSIVA** - ad un utenza specializzata, in grado di interagire con i prodotti in condizioni di sicurezza per le persone, per la macchina e per l'ambiente, interpretando un'elementare diagnostica dei guasti e delle condizioni di funzionamento anomale e compiendo semplici operazioni di verifica funzionale, nel pieno rispetto delle norme di sicurezza e salute vigenti.

Le informazioni riguardanti installazione, montaggio, smontaggio, manutenzione, aggiustaggio, riparazione ed installazione di eventuali accessori, dispositivi ed attrezzature, sono destinate - e quindi eseguibili - sempre ed in via esclusiva da personale specializzato avvertito ed istruito, o direttamente dall'ASSISTENZA TECNICA AUTORIZZATA, nel pieno rispetto delle raccomandazioni trasmesse dal costruttore e delle norme di sicurezza e salute vigenti.

I dispositivi non possono essere utilizzati all'aperto. Si deve sempre provvedere ad inserire i moduli all'interno di un contenitore a norme di sicurezza che rispetti le vigenti normative. La protezione di questo contenitore non si deve limitare ai soli agenti atmosferici, bensì anche a quelli meccanici, elettrici, magnetici, ecc.

Per un corretto rapporto coi prodotti, è necessario garantire leggibilità e conservazione del manuale, anche per futuri riferimenti. In caso di deterioramento o più semplicemente per ragioni di approfondimento tecnico ed operativo, consultare direttamente l'Assistenza Tecnica autorizzata.

Al fine di non incontrare problemi nell'uso di tali dispositivi, è conveniente che l'utente - PRIMA DI COMINCIARE AD OPERARE - legga con attenzione tutte le informazioni contenute in questo manuale. In una seconda fase, per rintracciare più facilmente le informazioni necessarie, si può fare riferimento all'indice generale e all'indice analitico, posti rispettivamente all'inizio ed alla fine del manuale.

La **grifo®** non garantisce che questo software soddisfi le richieste dell'utente, che la produzione non cessi o sia priva di errori o che tutti gli eventuali problemi siano risolti. La **grifo®** non é inoltre responsabile dei problemi causati dalle modifiche dell'hardware dei calcolatori o dei sistemi operativi che si possono verificare nel tempo.

grifo® si riserva il diritto di apportare cambiamenti e/o miglioramenti ai prodotti descritti in questo manuale, ed allo stesso manuale, in qualunque momemnto senza darne notizia.

Il programma descritto in questo manuale é coperto da diritti d'autore. Nè il programma nè alcuna sua parte possono essere analizzati, disassemblati o modificati in alcun modo, con qualunque mezzo, per qualunque scopo.

Tutti i marchi registrati che compaiono nel presente manuale sono proprietà dei relativi costruttori.

VERSIONE

Il presente manuale è riferito alla versione **2.1** della libreria **QTP 16Big** ed alle eventuali versioni successive. La validità delle informazioni riportate è quindi subordinata al numero di versione in uso e l'utente deve sempre verificarne la giusta corrispondenza. L'utente può ottenere il numero di versione in diversi modi:

- dalla scheda, come indicato nel relativo manuale utente **QTP 16Big**;
- tramite un apposito comando previsto nella libreria;
- nel nome del file di libreria ricevuto;

Normalmente la libreria **QTP 16Big** viene sempre fornita con l'ultima version disponibile, ma in caso di specifiche esigenze l'utente può richiedere anche una versione diversa, specificandolo in fase di ordine.

In questo manuale sono presenti delle informazioni relative ad altri programmi che costituiscono una parte integrante della libreria: ognuno di questi ha il proprio numero di versione che, quando necessario, viene presentato in questo manuale. Infine anche l'hardware é dotato di una propria versione.
In caso di necessità di assistenza tecnica é di fondamentale importanza che l'utente, oltre alla descrizione del problema, fornisca i numeri di versione di tutti i prodotti in uso.

Come ogni prodotto, anche la libreria **QTP 16Big** é soggetta a continue evoluzioni e modifiche, con l'intento di soddisfare nel modo migliore le nuove richieste dell'utenza e di eliminare gli eventuali problemi riscontrati. Di seguito viene quindi riportata una breve descrizione delle modifiche che il pacchetto ha subito, a seconda del numero di versione

Ver. 1.0 -> Versione di sviluppo e prove interne.

Ver. 1.3 -> Realizzata gestione per **QTP 12/R84**.

Ver. 2.1 -> Modificata per gestione della **QTP 16Big** con stampato versione **110705**.

Ogni eventuale aggiunta o miglioria che l'utente ritiene interessante, può essere proposta contattando direttamente la **grifo®**.

INDICAZIONI GENERALI

La libreria **QTP 16Big** coincide con un firmware appositamente sviluppato per consentire all'utente di sviluppare un **proprio programma applicativo** in modo comodo, veloce ed efficace. Con questa libreria la **QTP 16Big** opera come un potente **controllore completo di interfaccia operatore** che può quindi funzionare sia autonomamente che abbinato ad altri sistemi.
Una ricca e completa serie di **comandi**, che possono essere chiamati direttamente con i relativi **parametri** e **risultati**, consentono ad ogni utente di realizzare il programma applicativo che soddisfi al meglio le sue esigenze,con un **tempo di sviluppo** veramente **ridotto**. Tali comandi soddisfano le normali richieste dell'ambiente industriale e sono il frutto di una esperienza decennale in questo settore.

Si ricorda che in questo manuale si usa la sigla **QTP 16Big.LIB** per identificare la libreria in oggetto con un nome più conciso.

La **QTP 16Big.LIB** non é quindi un prodotto finito pronto per essere installato ma deve essere prima **specializzato** dall'utente. Questa specializzazione può essere effettuata con comodi ed economici **ambienti di sviluppo**, sia ad alto che a basso livello, e rendono la **QTP** un prodotto veramente flessibile e versatile. Infatti il programma applicativo dell'utente che la specializza consente di risolvere ogni problematica anche di alta complessità, ed allo stesso tempo consente di realizzare automazioni diverse, usando lo stesso hardware.
La maggioranza degli ambienti di sviluppo disponibili per la famiglia I51 possono usare la libreria **QTP 16Big** e mettono a disposizione comode modalità di **debug** del programma applicativo, riducendo ancora i tempi complessivi di preparazione. Tra questi si possono ricordare: **assemblatori**; compilatori **PASCAL** (SYS51PW); compilatori **C** (HTC-51, SYS51CW, µC/51); programmazione a **Contatti** (LadderWORK); compilatori **BASIC** (BASCOM-8051); ecc.

Il presente manuale fornisce le specifiche d'uso relative alla sola libreria mentre tutte le altre informazioni su connettori, configurazioni, comandi, collegamenti, montaggio ed installazione sono disponibili nel manuale utente della **QTP 16Big**. Tutte le indicazioni di quest'ultimo manuale che si differenziano per la **QTP 16Big.LIB**, sono riportate con le opportune modifiche, in paragrafi che hanno lo stesso nome. L'utente deve quindi prendere il manuale della **QTP 16Big** ed integrarlo con il manuale **Libreria QTP 16Big**, provvedendo a sostituire i paragrafi omonimi del primo manuale, con quelli del secondo.
La librearia può essere usata anche per la variante **QTP 16BigH**, ovvero nella versione del pannello operatore senza tastiera e LED: in questo caso il presente manuale dovrà essere integrato con il manuale d'uso corrispondente. L'utente deve ricordare che tutte le indicazioni **QTP 16Big** riportate nel presente manuale, si riferiscono indistintamente sia alla **QTP 16Big** che alla **QTP 16BigH**.

Le caratteristiche generali della libreria sono le seguenti:

- Facilmente utilizzabile per sviluppare numerose applicazioni di **interfaccia operatore**.
- **Riduce i tempi**, e quindi i **costi**, di **sviluppo** dell'applicativo.
- Utilizza una quantità minima di **risorse hardware**.
- Non usa la linea **seriale asincrona** che rimane a disposizione per collegamenti con altri sistemi, per il **debug** dell'applicativo e per la comunicazione di **console**.
- Caratterizzata da **ridotti tempi di esecuzione** in modo da poter affrontare anche problemi con tempistiche stringenti.
- Può essere **integrata** ed/od abbinata alla maggioranza dei **linguaggi di programmazione** disponibili sul mercato.

- Mantiene la **compatibilità** d'uso con gli altri firmware delle **QTP**.
- Include numerosi **comandi** equivalenti ad altrettante funzionalità.
- Prevede una **facile** modalità di **chiamata** ai comandi e di interscambio dei **parametri** e **risposte**.
- Copre le normali e più frequenti necessità del settore **industriale**.
- Per lo sviluppo necessita di un normale **PC** e non richiede nessun hardware aggiuntivo.
- Comoda selezione modalia' di partenza (**AUTORUN** o **DEBUG**).
- Per gli ambienti di sviluppo proposti da **grifo®** l'utente trova una completa descrizione sia delle **configurazioni** per effettuare l'**integrazione** che delle **direttive** per preparare il programma applicativo.
- Ampia **documentazione** e notevole serie di **esempi** sia in formato **sorgente** che **eseguibile**.
- **Nessuna licenza** o costo aggiuntivo. L'utente e' libero di realizzare tutte le applicazioni che desisdera.

Nel presente manuale é riportata una descrizione di tutte le caratteristiche della libreria **QTP 16Big** sufficienti per un suo uso.

FIGURA 1: ESEMPIO DI APPLICAZIONE CON QTP 16BIG.LIB

MATERIALE NECESSARIO

Viene di seguito riportata una breve descizione del materiale (hardware e software), necessario per operare con la libreria **QTP 16Big**:

QTP 16BIG

Coincide con il pannello operatore **QTP 16Big** o **QTP 16BigH** appartenenti al carteggio industriale **grifo®**. Come abbondantemente descritto nel manuale utente tale prodotto può essere fornito con due diversi tipi di display e con alcune opzioni aggiuntive. La libreria é in grado di gestire tutti questi modelli e qualsiasi combinazione di opzioni.
La scelta della configurazione del pannello deve comunque avvenire in relazione alle specifiche esigenze dell'applicazione che deve essere sviluppata.

PERSONAL COMPUTER

La libreria **QTP 16Big** necessita di un personal computer, che da ora in poi chiameremo **PC di sviluppo**, con le seguenti caratteristiche:

Personal Computer:	IBM compatibile (con CPU ≥ 486).
Memoria RAM:	Minimo 32M Bytes.
Sistema operativo:	WINDOWS 95, 98, ME, 2000, NT, XP
Monitor:	Colori.
Memorie di massa:	Lettore CD-ROM Hard Disk con almeno 30M Byte liberi.
Seriale:	RS 232, secondo specifiche V24
Mouse:	Microsoft compatibile con relativo driver installato.

CAVO DI COMUNICAZIONE SERIALE

Per alcune delle fasi previste dalla libreria é necessario effettuare un collegamento seriale tra una delle linee seriali del PC di sviluppo e la linea seriale della **QTP 16Big**. Tale collegamento necessita solo dei segnali di ricezione, trasmissione e massa (RxD, TxD e GND) e deve avvenire seguendo le normative V24 del C.C.I.T.T.

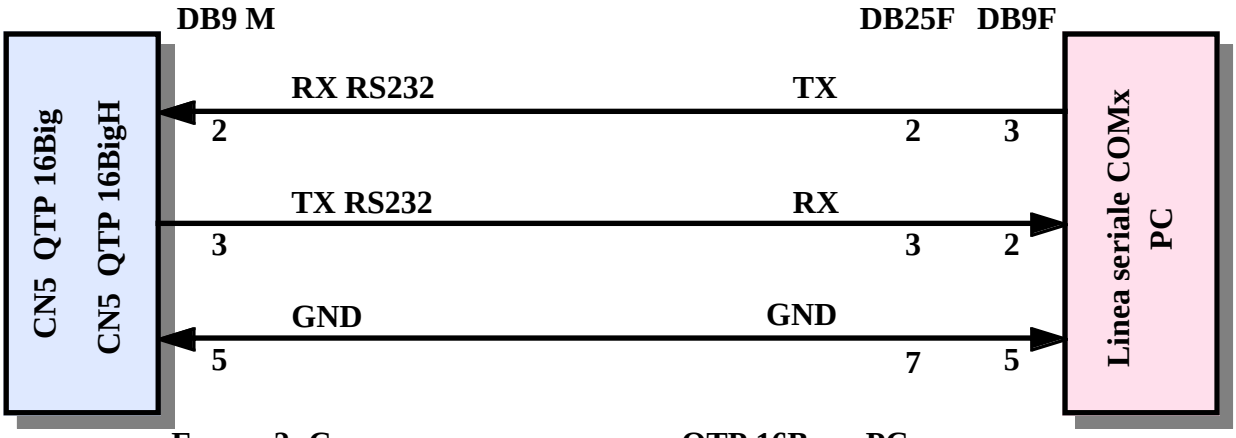


FIGURA 2: COLLEGAMENTO SERIALE TRA QTP 16BIG E PC DI SVILUPPO

Sul lato PC, qualora non sia disponibile alcuna linea di comunicazione seriale, si possono usare degli appositi convertitori che una volta aggiunti al PC in uso mettono a disposizione la linea seriale RS 232 richiesta. Tra questi si possono citare i convertitori USB<->RS 232, i convertitori ETHERNET<->RS 232, le schede multi I/O con seriali RS 232 aggiuntive, ecc. Naturalmente questi dispositivi possono essere usati solo se correttamente installati sia dal punto di vista hardware che software, secondo le indicazioni della casa costruttrice.
Al fine di velocizzare la fase di collegamento ed eliminare la necessità di dover realizzare un cavo di comunicazione, la **grifo®** é in grado di offrire degli accessori già pronti, ad esempio ordinando il **CCR 9+9R**.

SOFTWARE DI LAVORO

Assieme all'hardware descritto per operare con la libreria **QTP 16Big** sono necessari dei software di lavoro con cui sviluppare e mettere a punto il programma applicativo. Tale software é organizzato sotto forma di appositi pacchetti e può essere suddiviso in alcuni gruppi principali, come di seguito descritto.

PROGRAMMA SCRITTURA FLASH EPROM: FLIP

Come descritto nei capitoli seguenti la libreria **QTP 16Big** deve essere salvata nella FLASH EPROM del microcontrollore assieme al programma applicativo utente. Questo salvataggio avviene tramite una programmazione ISP (In System Programming) che riduce i costi ed i tempi di sviluppo dell'applicazione, infatti elimina la necessità di usare EPROM esterne, programmatori, cancellatori, simulatori, ecc.
La programmazione ISP avviene tramite il PC di sviluppo che eseguendo un apposito programma di gestione, denominato **FLIP** (FLexible In system Programming), interagisce con un Boot loader presente sul microcontrollore e consente di leggere, cancellare, verificare, programmare sia la memoria FLASH che quella EEPROM. Il tutto avviene tramite un collegamento tra PC e **QTP 16Big.LIB** normalmente effettuato con una linea seriale RS 232 (vedi figura 2) oppure, in alternativa,con linea CAN (per quest'ultima possibilità contattare direttamente la **grifo®**).
Il programma **FLIP** é fornito nel materiale ricevuto con l'ordine della libreria **QTP 16Big**, ma può essere anche scaricato gratuitamente dal sito della ATMEL, seguendo il percorso:

"Products | Microcontrollers | 8051 Architecture | Tools & Software | FLIP"

PROGRAMMA EMULAZIONE TERMINALE SERIALE

Coincide con un generico programma di comunicazione in grado di gestire una classica emulazione terminale, con un protocollo fisico di comunicazione impostabile. Tale programma viene usato come console per i programmi demo forniti e deve rappresentare sul monitor tutto quanto ricevuto dalla linea seriale e trasmettere, sulla stessa seriale, quanto premuto sulla tastiera.
A questo scopo si ricorda il **GET51** sviluppato dalla **grifo®**, il famoso **HYPERTERMINAL** di Windows, od i diffusi programmi **CROSS TALK**, **PROCOMM**, **BITCOMM**, **TERMINAL**, ecc. realizzati da terze parti.

AMBIENTE DI SVILUPPO PROGRAMMA APPLICATIVO

Prima di essere programmato sulla **QTP 16Big.LIB**, il programma applicativo dell'utente deve essere generato. A questo scopo si possono convenientemente usare degli appositi ambienti di sviluppo che consentono di scrivere il programma comodamente sul PC di sviluppo e di trasformarlo nel codice macchina usato dal microcontrollore.
In generale la scheda può sfruttare tutte le risorse software per il microcontrollore montato, ovvero i numerosi pacchetti ideati per la famiglia I51, sia ad alto che a basso livello. Gli ambienti di sviluppo software forniti dalla **grifo®** sono sempre accompagnati da esempi (in formato sorgente ed eseguibile), librerie con console ridirezionabile, file di intestazione ed accessori che effettuano l'integrazione della libreria e la rendono pronta all'uso.
Tra questi ricordiamo:

HI TECH C 51: Cross compilatore per file sorgenti scritti in linguaggio C. E' un potente pacchetto software che tramite un comodo IDE permette di utilizzare un editor, un compilatore C (floating point), un assemblatore, un ottimizzatore, un linker e un remote debugger. Sono inoltre inclusi i source delle librerie.

SYS51CW: Cross compilatore per programmi scritti in C, disponibile in ambiente WINDOWS con un comodo IDE che mette a disposizione: editor, compilatore C, assemblatore, ottimizzatore, linker, librerie ed un debugger simbolico remoto.

SYS51PW: Cross compilatore per programmi scritti in PASCAL, disponibile in ambiente WINDOWS con un comodo IDE che mette a disposizione: editor, compilatore PASCAL, assemblatore, ottimizzatore, linker, librerie ed un debugger simbolico remoto.

DDS MICRO C 51: E' un comodo pacchetto software, a basso costo, che tramite un completo IDE permette di utilizzare un editor, un compilatore C (integer), un assemblatore, un linker e un remote debugger abbinato ad un monitor. Sono inclusi i sorgenti delle librerie ed una serie di utility.

BASCOM 8051: Cross compilatore a basso costo per files sorgenti scritti in BASIC, disponibile in ambiente WINDOWS con un comodo IDE che mette a disposizione un editor, il compilatore ed un simulatore molto potente per il debugger del sorgente. Comprende molti modelli di memoria, svariati tipi di dati e numerose istruzioni dedicate alle tipiche risorse hardware usate nell'automazione industriale.

µC/51: E' un comodo pacchetto software, a basso costo, che tramite un completo IDE permette di utilizzare un editor, un compilatore ANSI C, un assemblatore, un linker e un remote debugger configurabile da utente, a livello sorgente. Sono inclusi i sorgenti delle librerie fondamentali e del remote debugger, alcuni esempi di utilizzo e vari programmi di utility.

LADDER WORK: E' un semplice sistema per creare programmi di automazione con la conosciuta e diffusa logica a contatti. Include un editor grafico che consente di posizionare e collegare i componenti hardware della scheda (input, output, contatori, A/D, ecc) come su uno schema elettrico e di definne le proprietà, un efficiente compilatore che converte lo schema in codice eseguibile ed utility per il download di tale codice verso la scheda. Il tutto integrato in un comodo IDE per Windows. Viene fornito sotto forma di CD che comprende esempi e manuale d'uso e relativa chiave di abilitazione.

DESCRIZIONE LIBRERIA QTP 16BIG

In questo capitolo vengono descritte tutte le caratteristiche della libreria **QTP 16Big** relative sia al suo uso a bordo del pannello operatore, che alla generazione del programma applicativo. Se l'utente desidera velocizzare il primo utilizzo del pacchetto, si consiglia di seguire i passi dei capitoli COME INIZIARE... che, quando é necessario, rimandano ai paragrafi di questo capitolo.

CONFIGURAZIONE QTP 16BIG

Per poter usare la libreria **QTP 16Big** lo stesso prodotto deve essere opportunamente configurato. Tale configurazione si rivolge principalmente alla configurazione della linea seriale asincrona di comunicazione su cui si basano tutte le operazioni di scaricamento, prova e salvataggio del programma applicativo.
In generale la configurazione necessaria é quella che setta la linea seriale in RS 232, ovvero:

J3, J4	->	non connessi
J1	->	da connettere e non connettere durante l'uso
IC11	->	driver MAX 202

Mentre tutti gli altri jumpers non sono significativi e possono essere settati a seconda delle proprie esigenze di lavoro. Qualora la linea seriale asincrona in RS 232 sia una configurazione inaccettabile per il programma applicativo utente, si possono individuare delle soluzioni alternative, contattando direttamente la grifo®.
Si ricorda che la configurazione del jumper J1 deve essere variata frequentemente durante lo sviluppo del programma applicativo; se l'uso di un normale jumper a cavalliere risulta scomodo, si può usare in alternativa un connettore per remotarlo, usando gli appositi accessori descritti nel manuale **QTP 16Big**. In quest'ultimo manuale si trovano anche le figure che identificano la posizione degli zoccoli e dei jumpers descritti.

BUFFER DI COMUNICAZIONE

La **QTP 16Big.LIB** é dotata di due buffer di comunicazione che rendono possibile il passaggio parametri e risultati dei comandi forniti dal programma applicativo utente
Il primo buffer é di ricezione, è lungo **24 bytes**, memorizza i dati ricevuti dal programma utente e viene esaminato e svuotato al termine del comando in corso. Visto che una volta completata la ricezione del comando, la libreria lo esegue immediatamente, non si possono presentare i problemi di riempimento e traboccamento della **QTP 16Big**. Il programma applicativo non necessita quindi di alcun ritardo antitraboccamento.
Il secondo buffer é di trasmissione, é lungo **20 bytes**, memorizza i dati che la libreria deve restituire al programma applicativo utente e viene quindi riempito con i codici dei tasti premuti e con le risposte dei comandi. Visto che i dati rimangono nel buffer di trasmissione fino a quando il programma utente non li richiede, tale buffer é destinato a riempirsi ed in caso di riempimento tutti i dati sucessivi vengono persi. Quindi il programma applicativo utente deve gestire la ricezione dalla libreria **QTP 16Big** almeno in due situazioni: prima di fornire comandi con risposte (per svuotare il buffer per le stesse risposte) e periodicamente (per prelevare gli eventuali tasti premuti).

INTEGRAZIONE ED USO DELLA LIBRERIA

La librearia **QTP 16Big** é stata progettata prefissando i seguenti obiettivi:

- poter essere abbinato a tutti i linguaggi di programmazione disponibili;
- ridurre al minimo le risorse hardware usate;
- mantenere la compatibilità d'uso con gli altri firmware delle **QTP**;
- prevedere un facile modalità di chiamata ai comandi ed interscambio parametri;
- coprire le normali e più frequenti necessità del settore industriale;

che hanno definito le modalità di integrazione ed uso dello stesso firmware, all'interno del programma applicativo dell'utente.

Per chiarezza si ricorda che l'utente della libreria **QTP 16Big** deve avere una conoscenza base del microcontrollore usato e della realizzazione di software embedded in quanto la sucessiva documentazione non fornisce, ma usa, queste informazioni. Tali conoscenze possono essere acquisite dalla lettura dei data sheet, riportato in APPENDICE A del manuale.
In questo paragrafo sono riportate tutte le informazioni generali sull'integrazione e l'uso della libreria **QTP 16Big** che possono essere utilizzate da ogni utente, con qualsiasi ambiente di sviluppo fornito da **grifo®** o da terze parti.

Per integrare ed usare la libreria sono necessari alcuni strumenti hardware e software che vengono opportunamente indicati nella sucessiva descrizione. La documentazione completa di tali strumenti é disponibile all'interno degli stessi e non viene quindi riportata in questo manuale.

Concludendo le operazioni necessarie per integrare ed usare la libreria sono:

a) Installare l'ambiente di sviluppo prescelto per realizzare il programma applicativo, sul PC di sviluppo. In generale la scheda può sfruttare tutti gli ambienti per il microprocessore montato, ovvero i numerosi pacchetti ideati per la famiglia 51, sia ad alto che a basso livello (assemblatori, compilatori, interpreti, ecc.). Tutti i pacchetti di sviluppo software forniti dalla **grifo®** sono sempre accompagnati dagli elementi che integrano completamente il firmware rendendolo pronto all'uso.

b) Installare l'ambiente di programmazione ISP (In System Programming) sul PC di sviluppo, ovvero il programma FLIP che comunicando con il Boot loader del microcontrollore, attraverso la porta seriale, permette di leggere, cancellare e riscrivere la memoria FLASH EPROM.
La programmazione ISP riduce i costi ed i tempi di sviluppo dell'applicazione, infatti elimina la necessità di usare EPROM esterne, programmatori, cancellatori, ecc. Per ulteriori informazioni sulla programmazione ISP si prega di consultare la specifica documentazione tecnica rilasciata dalla ATMEL.

c) Predisporre l'ambiente di sviluppo in modo che nel programma applicativo generato siano preservate le risorse hardware usate dalla libreria; come illustrato nelle figure 3, 4 e nel sucessivo paragrafo RISORSE USATE DALLA LIBRERIA, il programma applicativo non deve usare l'area finale della FLASH, alcune aree di RAM interna, un timer counter, la EEPROM di bordo, ecc.

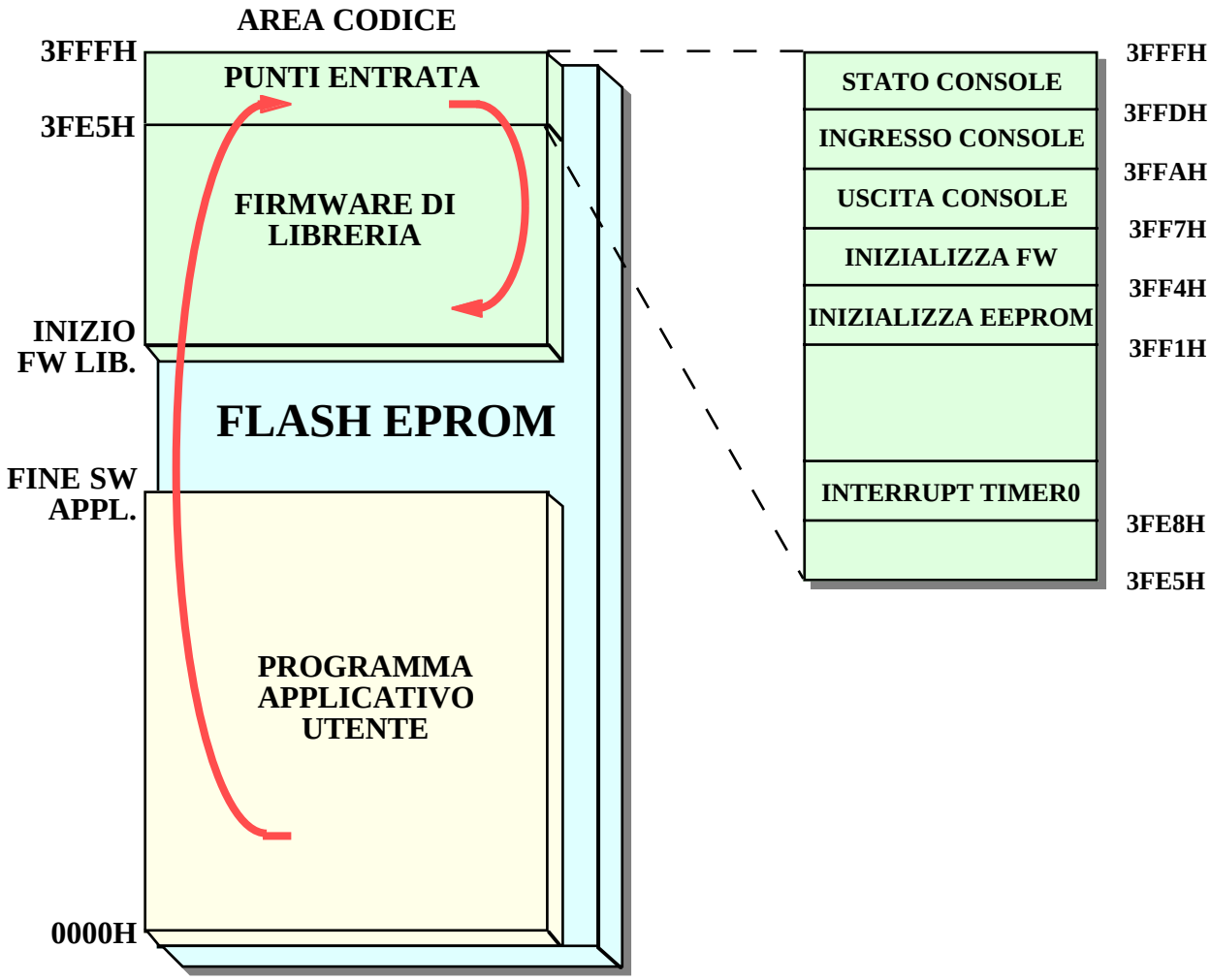


FIGURA 3: ORGANIZZAZIONE AREA CODICE CON LIBRERIA

d) Fisicamente la libreria coincide con un codice eseguibile che deve essere salvato al termine della area codice del microcontrollore come indicato in figura 3. Tale codice viene fornito nel file QTP16Bxx.HEX che, grazie al suo formato HEX, può essere utilizzato direttamente per la programmazione della FLASH. In quest'ultima memoria oltre al firmware descritto si dovrà salvare anche il codice eseguibile del programma applicativo utente che deve essere naturalmente allocato all'inizio dell'area codice, in modo che venga immediatamente eseguito a seguito di un'accensione o di un reset. Il passaggio dal programma applicativo alla libreria é effettuato tramite un'opportuna tabella di entrata, allocata ad indirizzi fissi, che funge da ponte di collegamento tra i due codici presenti.
La scelta degli indirizzi di allocazione delle tre aree in FLASH é stata effettuata attentamente in modo da avere il massimo spazio libero per il programma applicativo, avere dei punti di entrata che rimangono inalterati anche in caso di aggiornamenti ed ampliamenti della libreria ed avere punti di entrata coincidenti nelle librerie di **QTP** diverse. Così facendo l'utente può usare una nuova versione di libreria od una **QTP** diversa, semplicemente riprogrammandola nella FLASH, senza intervenire sul suo programma applicativo.

e) Il valore di **INIZIO FW LIB.** é stabilito dalla libreria stessa e quindi varia al variare della sua versione; attualmente con la versione 2.1 é fissato a **2C00H** e comunque può essere facilmente determinato caricando il file *QTP16Bxx.HEX* (dove xx corrisponde al numero di versione) ed esaminando il suo indirizzo di inizio.

L'utente a seguito di ogni realizzazione del programma applicativo deve sempre verificare che il relativo indirizzo di fine **FINE SW APPL**. sia inferiore all'**INIZIO FW LIB.** ovvero che i due codici non si sovrappongano. Questa verifica può essere effettuata facilmente infatti normalmente tutti gli ambienti di sviluppo (assemblatori, compilatori, linguaggi, ecc.) forniscono indicazioni sulle dimensioni del codice generato e sarà quindi sufficiente confrontare tali indicazioni con il valore di **INIZIO FW LIB.** descritto prima.

f) Ridirezionare il vettore di risposta all'interrupt TIMER0 del microcontrollore, al corrispondente punto d'entrata della libreria indicato in figura 3. La ridirezione descritta deve essere realizzata seguendo le regole dell'ambiente di sviluppo e normalmente coincide con un'istruzione di salto assoluto (ad esempìio LJMP 3FE8H) all'indirizzo di entrata, posta nella procedura di risposta al relativo interrupt. Si ricorda che la ridirezione dell'interrupt TIMER0 é indispensabile per ottenere un completo funzionamento della libreria, in particolare per tutti i processi basati sul tempo come la scansione tastiera, attivazione temporizzata delle risorse, intermittenza, shift messaggi, rappresentazione orologio, ecc.

g) Al fine di semplificare l'uso della libreria, di tutti i suoi comandi ed il passaggio parametri e risultati sono state previste tre procedure, con altrettanti punti di entrata, con le seguenti caratteristiche:

STATO CONSOLE: restituisce lo stato di presenza di un dato che la libreria deve fornire al programma applicativo; il dato può essere il codice di un tasto premuto oppure la risposta ad un comando inviato. La procedura non ha variabili d'ingresso ed una sola variabile d'uscita salvata nell'Accumulatore che coincide con il numero di caratteri pronti per essere forniti al programma applicativo. Tale numero di caratteri funziona anche da stato infatti se azzerato non vi sono dati e viceversa.

INGRESSO CONSOLE: attende la disponibilità di un dato che la libreria fornisce al programma applicativo e lo restituisce; anche per questa procedura il dato può essere il codice di un tasto premuto oppure la risposta ad un comando inviato. La procedura non ha variabili d'ingresso ed una sola variabile d'uscita salvata nell'Accumulatore, che coincide con il dato descritto.

USCITA CONSOLE: invia un dato dal programma applicativo alla libreria; il dato può essere un carattere da rappresentare sul display oppure un comando da inviare od i suoi eventuali parametri. La procedura ha una sola variabile d'ingresso salvata nell'Accumulatore che coincide con il dato descritto e non ha variabili d'uscita.

Nel programma applicativo la maggioranza delle funzioni offerte dalla libreria vengono usate con le tre procedure descritte; per l'utente é conveniente ridirezionare le procedure di console dell'ambiente di sviluppo scelto (che hanno già una struttura compatibile) su queste procedure. Questo metodo spiega il suffisso CONSOLE che é stato attribuito alle stesse procedure ed allo stesso tempo fornisce una notevole semplificazione ed una impareggiabile potenzialità d'uso. Infatti le istruzioni ad alto livello *PRINT*, *PRINTF*, *PUTS*, *KBHIT*, *SCANF*, *INPUT*, ecc. di un ambiente di sviluppo in C o BASIC, chiamano automaticamente le tre procudure di entrata della libreria consentendo di utilizzare tutte le loro possibilità. Ad esempio l'istruzione C:

printf("Pezzi prodotti=%4d",npezzi);

rappresenta sul display la stringa *Pezzi prodotti=* seguita dal valore della variabile *npezzi*, formattata a 4 cifre decimali.

Le tre procedure descritte usano e quindi modificano, la maggioranza dei registri del microprocessore. Qualora l'ambiente di sviluppo non consenta queste variazioni, l'utente deve provvedere a salvarli e ripristinarli prima e dopo ogni loro chiamata.

g) A seguito di un reset o di un'accensione il programma applicativo utente deve predisporre la libreria alle sucessive operazioni. Per queste inizializzazioni sono disponibili due procedure, con altrettanti punti di entrata, con le seguenti caratteristiche:

INIZIALIZZA FW: effettua tutte le operazioni di inizializzazione necessarie come:
- settaggio variabili;
- azzeramento buffer;
- disattivazione di buzzer, LEDs di stato, uscite digitali;
- disabilitazione e disattivazione sveglia;
- inizializzazione e cancellazione display;
- settaggio cursore blocco lampeggiante, nella posizione di Home;
- caricamento dei caratteri utente;
- visualizzazione eventuale rappresentazione automatica d'accensione
- settaggio del keyclick impostato;
- attivazione scansione tastiera;
- attivazione funzioni temporali;
- ecc.

La procedura non ha variabili d'ingresso ed uscita e deve essere sempre chiamata prima di ogni altra funzione con la libreria.

INIZIALIZZA EEPROM: inizializza la EEPROM di base con i dati di default descritti nel paragrafo DATI IN EEPROM del manuale **QTP 16Big** e poi effettua tutte le operazioni di inizializzazione descritte per la procedura INIZIALIZZA FW. La procedura ha due variabili d'ingresso e non ha variabili d'uscita; il passaggio delle variabili d'ingresso é effettuata tramite la seguente tecnica:

registro *Accumulatore* -> funzione relé RL2 equivalente all'uscita digitale NO OUT2
- 0 -> uscita utente
- 1 -> uscita sveglia

registro *B* -> dimensione EEPROM opzionale
- 0 -> nessuna EEPROM opzionale
- 1 -> EEPROM opzionale da 16K Bytes (**QTP 16Big-xx.LIB.EE128**)
- 2 -> EEPROM opzionale da 32K Bytes (**QTP 16Big-xx.LIB.EE256**)
- 3 -> EEPROM opzionale da 64K Bytes (**QTP 16Big-xx.LIB.EE512**)

Si ricorda che la durata dell'esecuzione di questa procedura é di circa 25 secondi. La procedura INIZIALIZZA EEPROM deve essere chiamata una sola volta per salvaguardare la durata dell'EEPROM su cui scrive: i suoi usi tipici sono quelli in corrispondenza della prima installazione, oppure quando si devono ripristinare i settaggi iniziali, a causa di modifiche indesiderate.

Normalmente entrambe le procedure di inizializzazione devono essere chiamate una sola volta all'inizio del programma applicativo, usando le modalità dell'ambiente di sviluppo scelto; queste normalmente coincidono con una istruzione di chiamata assoluta (ad esempio LCALL 3FF4H) all'indirizzo di entrata, preceduta da un eventuale settaggio delle variabili (=registri) d'ingresso.

h) Una volta realizzato il programma utente che usa la libreria con le caratteristiche descritte nei punti precedenti lo si deve salvare nella FLASH EPROM della **QTP 16Big.LIB** assieme alla stessa libreria, come descritto nel paragrafo PROGRAMMAZIONE FLASH EPROM.

i) A questo punto la **QTP 16Big.LIB** é completa e pronta ad essere usata e testata all'interno dell'applicazione utente. Il debug del programma ottenuto può essere effettuato con gli strumenti messi a disposizione dall'ambiente di sviluppo usato, ricordando che la linea seriale (non usata dalla libreria) é un ottimo candidato per questa funzione.

RISORSE USATE DALLA LIBRERIA

La libreria dispone di comandi che consentono di gestire facilmente la maggioranza delle risorse della scheda come display, contrasto, tastiera, buzzer, LEDs, uscite a relé, memorie EEPROM e SRAM tamponata, orologio, interfaccia I2C BUS, ecc. Tali comandi però utilizzano una serie di risorse hardware aggiuntive della **QTP 16Big** che sono brevemente descritte in questo paragrafo, assieme alle conseguenti limitazioni d'uso, da parte del programma applicativo utente:

Area codice in FLASH EPROM del microcontrollore: coincide con la porzione finale della FLASH EPROM, già descritta nei punti (d), (e) del paragrafo precedente. Su tale area viene salvato il codice della libreria e non deve essere assolutamente usata dal programma applicativo, pena il malfunzionamento di tutto il sistema. Alcuni ambienti di sviluppo (come il BASCOM 8051) possono essere configurati per avvisare l'utente in caso di generazione di un codice che supera un limite impostato (**FINE SW APPL.** >= **INIZIO FW LIB.**).

Aree dati in RAM interna del microcontrollore: coincide con due aree di RAM interna di cui la prima allocata nell'area ad accesso diretto e la seconda ad accesso indiretto, in cui sono salvate tutti i flag, le variabili ed i buffer della libreria. Queste aree sono allocate agli indirizzi fissi descritti in figura 4 e non devono essere assolutamente usate dal programma applicativo, pena il malfunzionamento di tutto il sistema.

La salvaguardia d'uso di queste memorie é realizzata seguendo le regole dell'ambiente di sviluppo usato e normalmente avviene con direttive per il compilatore, settaggi dell'eventuale progetto, dichiarazione di variabili che vengono solo allocate ma mai usate, utilizzo di adeguati codici di partenza (start up), ecc. Le finestre con i risultati della generazione del programma utente, normalmente proposte dall'ambiente di sviluppo, consentono di verificare facilmente la salvaguardia di tali aree, evitando di scoprire i conseguenti malfunzionamenti in fase di prova.

Si ricorda che la scelta d'uso della memoria della libreria é stata effettuata cercando di lasciare comunque a disposizione tutti i tipi di memoria del microcontrollore, ovvero: 4 byte indirizzabili a bit per un totale di 32 bit utente, 48 byte ad accesso diretto, 63 byte ad accesso indiretto ed infine 256 byte di RAM esterna. Inoltre la completezza dei comandi offerti dal firmware riduce drasticamente la necessità di memoria e variabili nel programma applicativo utente, tanto che spesso si riduce allo stack ed a poche variabili di lavoro.

Area di stack del microcontrollore: la libreria non ha un proprio stack e quindi usa quello del programma applicativo. L'utente in fase di configurazione dell'ambiente di sviluppo scelto deve tenere conto della profondità di stack necessaria al firmware, che nella condizione peggiore può arrivare a 27 bytes.

Contatore temporizzatore TIMER0 del microcontrollore: per gestire tutti i processi temporali la libreria necessita di un interrupt periodico generato dal TIMER0 del microcontrollore. Il programma applicativo utente non può usare questa risorsa, deve ridirezionare il relativo vettore d'interrupt del microcontrollore (000BH) al punto di entrata (3FE8H) e deve ricordare che una volta inizializzato, il firmware risponde a questo interrupt ogni 2,5 msec con un conseguente rallentamento nell'esecuzione. La riduzione delle prestazioni del programma applicativo dipende dai comandi usati e dai processi in corso ed in alcuni casi può arrivare a decine di msec, come opportunamente segnalato nella descrizione degli stessi comandi.

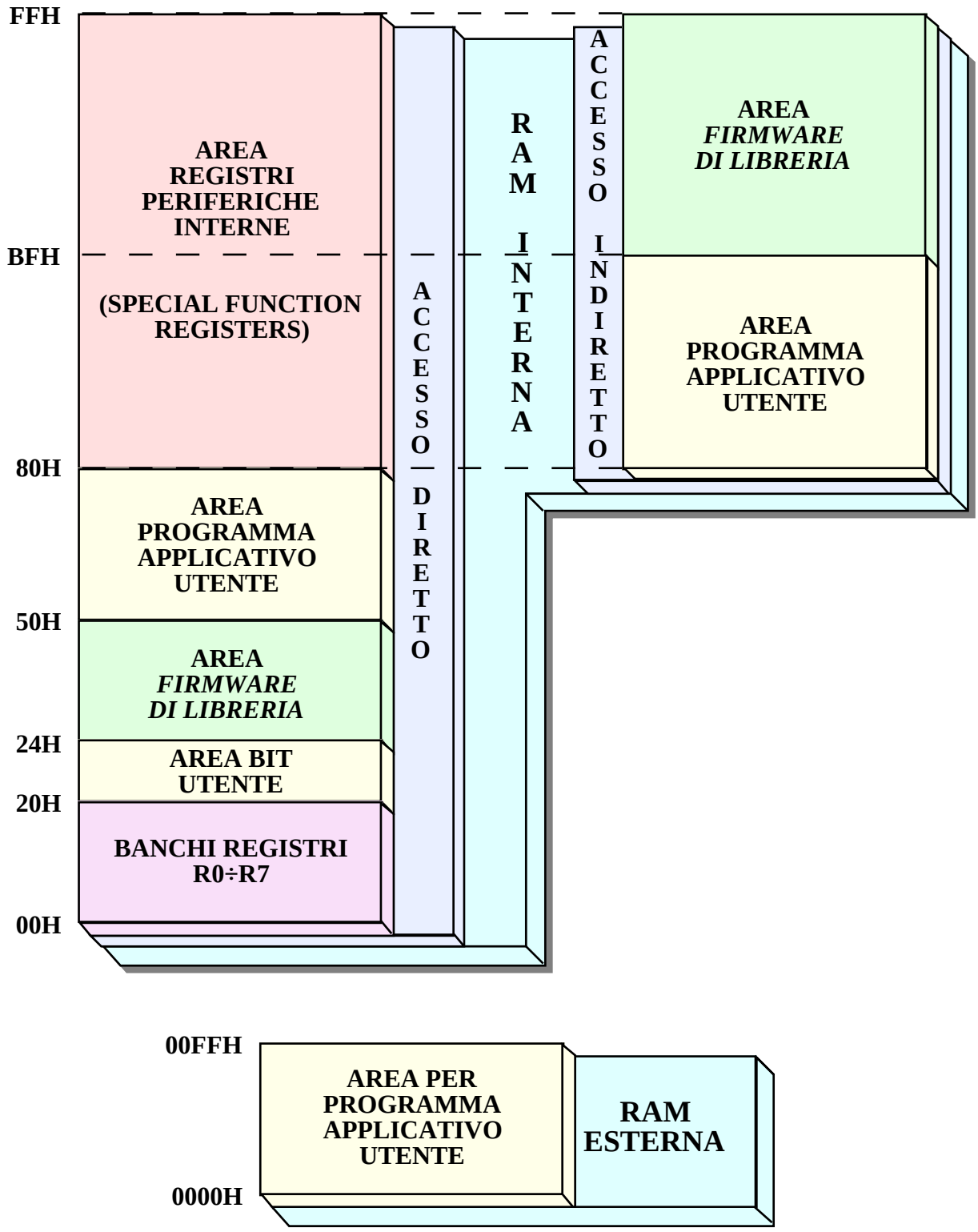


FIGURA 4: USO RAM CON LIBRERIA

PROGRAMMAZIONE FLASH EPROM

Come descritto nei paragrafi precedenti la libreria deve essere salvata nella FLASH EPROM del microcontrollore assieme al programma applicativo utente, come illustrato in figura 3. Questo salvataggio avviene tramite una programmazione ISP (In System Programming) che riduce i costi ed i tempi di sviluppo dell'applicazione.
La programmazione ISP avviene tramite il PC di sviluppo che, eseguendo un apposito programma di gestione denominato FLIP (FLexible In system Programming), interagisce con un Boot loader presente sul microcontrollore e consente di leggere, cancellare, verificare, programmare sia la memoria FLASH che quella EEPROM. Il tutto avviene tramite un collegamento tra il PC e la **QTP 16Big.LIB** normalmente effettuato con linea seriale RS 232 oppure, in alternativa,con linea CAN (per quest'ultima possibilità contattare direttamente la **grifo®**).
Come descritto nel manuale della **QTP 16Big** il jumper J1 seleziona la modalità operativa tra le due disponibili:

Non connesso	->	Modalità RUN
Connesso	->	Modalità DEBUG

In modalità RUN a seguito di un'accensione parte sempre il programma applicativo salvato in FLASH indipendentemente dalle condizioni esterne, mentre in modalità DEBUG l'accensione provoca l'esecuzione del Boot loader del microcontrollore e consente quindi la programmazione ISP.
Si ricorda che la configurazione del jumper J1 deve essere variata frequentemente durante lo sviluppo del programma applicativo; se l'uso di un normale jumper a cavalliere risulta scomodo, si può usare in alternativa un connettore per remotarlo, usando gli appositi accessori descritti nel manuale **QTP 16Big**.
Nei punti seguenti vengono illustrate le istruzioni che l'utente deve compiere per programmare correttamente la FLASH della **QTP 16Big.LIB**:

01) Installare il programma FLIP prelevandolo dal sito ATMEL o dal CD **grifo®** ricevuto.
02) Impostare modalità DEBUG, ovvero collegare il jumper J1.
03) Chiudere ogni programma che possa usare la linea seriale COMx del PC di sviluppo.
04) Fornire alimentazione alla **QTP 16Big**.
05) Lanciare il programma FLIP installato al punto 1.
06) Premere il primo pulsante in alto a sinistra, scegliere il microcontrollore da programmare nella finestra che appare e quindi premere OK. Tale selezione deve essere effettuata a seconda della **QTP** ordinata, ovvero *T89C5115* in caso di **QTP 16Big** e *T89C51CC02* in caso di **QTP 16Big.CAN**.

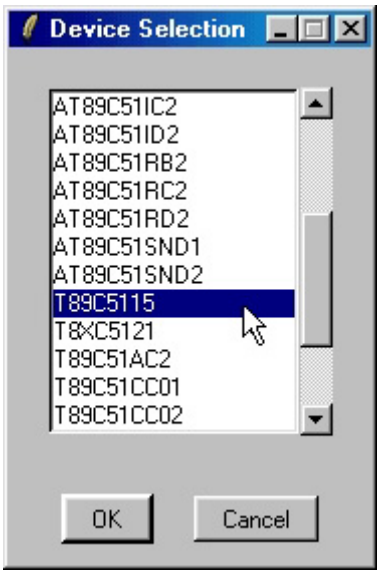


FIGURA 5: FINESTRA SETTAGGIO FLIP (1 DI 4)

07) Impostare le modalità di comunicazione per la programmazione ISP della **QTP** ovvero premere il secondo pulsante in alto da sinistra, scegliendo in sequenza: RS 232, la *porta* seriale del PC, *115200 Baud* e quindi premendo il pulsante Connect. A questo punto il FLIP instaura la comunicazione con il Boot loader del microcontrollore e compila una serie di dati nella sua finestra principale. Se la comunicazione non si instaura e dopo circa 20 secondi appare la finestra *Timeout Error*, potete provare ad: abbassare la velocità di comunicazione da 115200 a 19200 Baud; ed a ripetere i punti 2÷7.

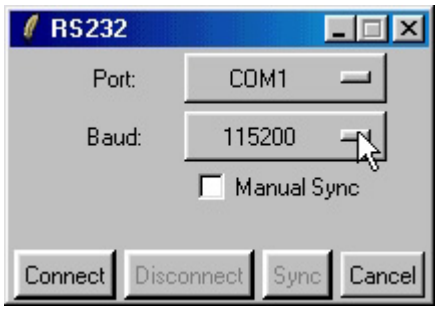


FIGURA 6: FINESTRA SETTAGGIO FLIP (2 DI 4)

08) Aprire la finestra *Buffer | Options* settare a *NO* la voce *Reset Buffer Before Loading*, selezionare la voce *Whole buffer* in modo da garantire il sucessivo corretto caricamento dei file da programmare e confermare le scelte con il tasto *0K*.

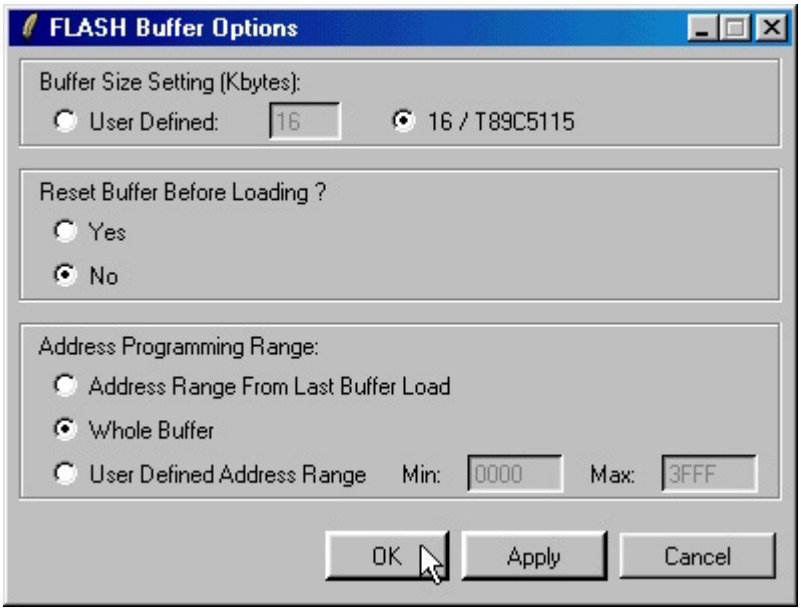


FIGURA 7: FINESTRA SETTAGGIO FLIP (3 DI 4)

09) Caricare i due file da salvare nella FLASH EPROM ovvero la libreria *QTP16Bxx.HEX* ed il programma applicativo *<file utente>.HEX*, effettuando due volte le seguenti operazioni: premere il nono pulsante da sinistra e selezionare il file tramite la finestra di dialogo che appare.
10) Spuntare tutte le caselle del riquadro *Operations Flow* come in figura 8, in modo che il FLIP esegua sequenzialmente le quattro operazioni di: cancellazione, verifica di cancellazione, programmazione e verifica di programmazione.
11) A questo punto assicurarsi che la finestra principale del FLIP si presenti come in figura 8, in particolare per le voci *Size:*, *X2*; *Device SSB* e *BSB / EB / SBV*.

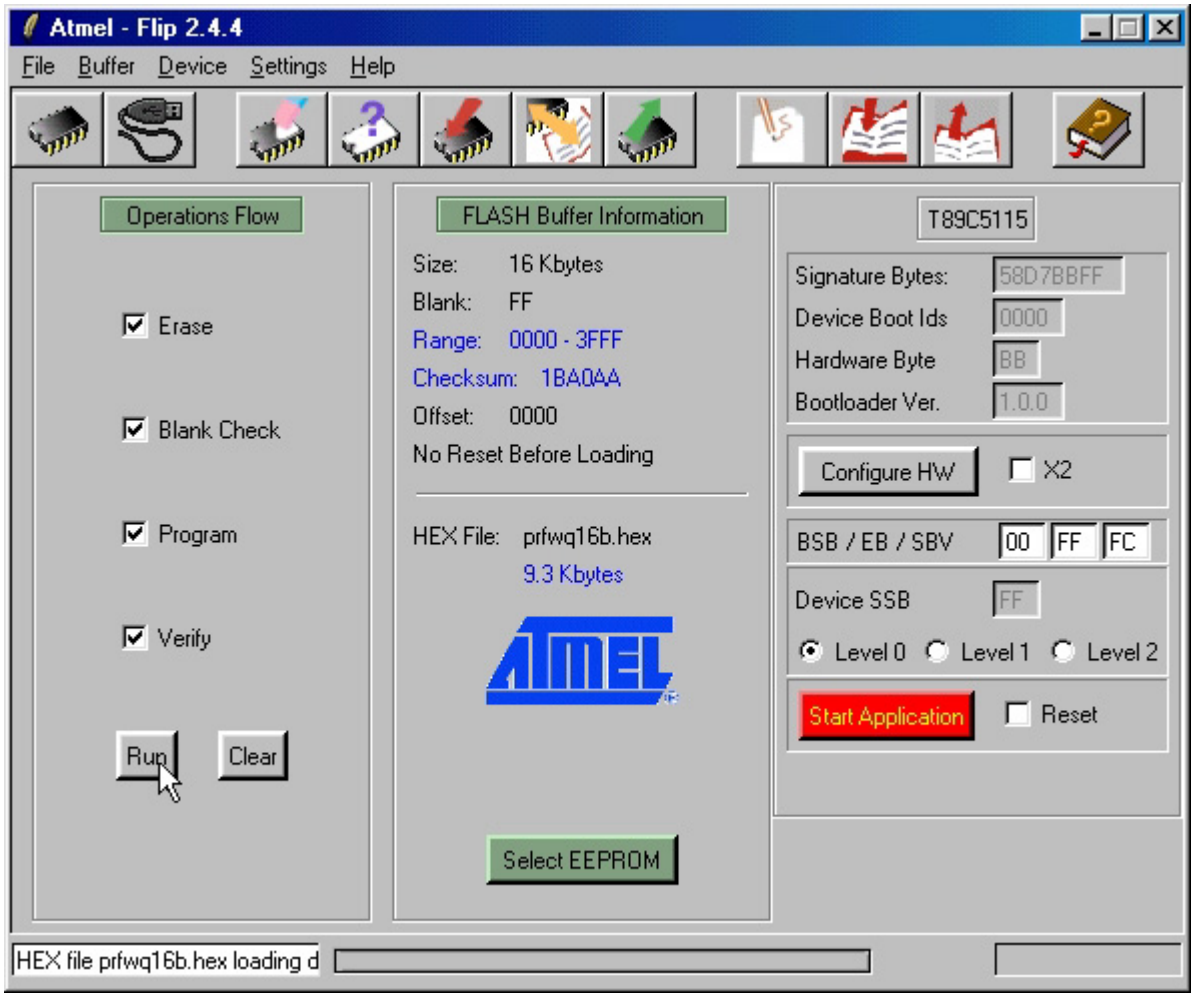


FIGURA 8: FINESTRA SETTAGGIO FLIP (4 DI 4)

12) Premere il pulsante *Run* nella finestra principale per avviare le operazioni ISP impostate.
13) Attendere la fine delle operazioni ISP. Nel riquadro in basso a sinistra si può vedere l'operazione in corso affiancata da una barra a scorrimento che indica il suo stato di avanzamento; le caselle di spunto diventano rosse durante l'esecuzione e poi verdi man mano che la rispettiva operazione viene completata. Si deve quindi aspettare che la casella *Verify* diventi verde.
14) A questo punto la FLASH é programmata ed il FLIP può essere chiuso.

Al fine di velocizzare le operazioni sopra descritte, che devono essere ripetute ad ogni prova del programma applicativo, si può conveniente usare la modalità BatchISP del FLIP corredata di un adeguato file di comando, con le seguenti istruzioni:

-device T89C5115
-hardware RS232
-port COM1
-baudrate 115200
-operation

(segue su colonna a destra)

memory FLASH
erase F
loadbuffer "QTP16B21.HEX"
loadbuffer "<file utente>.HEX"
addrange 0x0000 0x3FFF
program
verify

Nei paragrafi COME INIZIARE.... sono disponibili esempi di programmazione della FLASH, corredati da fotografie esplicative; per ulteriori informazioni sulla programmazione ISP e sull'uso del programma FLIP si prega di consultare la specifica documentazione tecnica rilasciata dalla ATMEL.

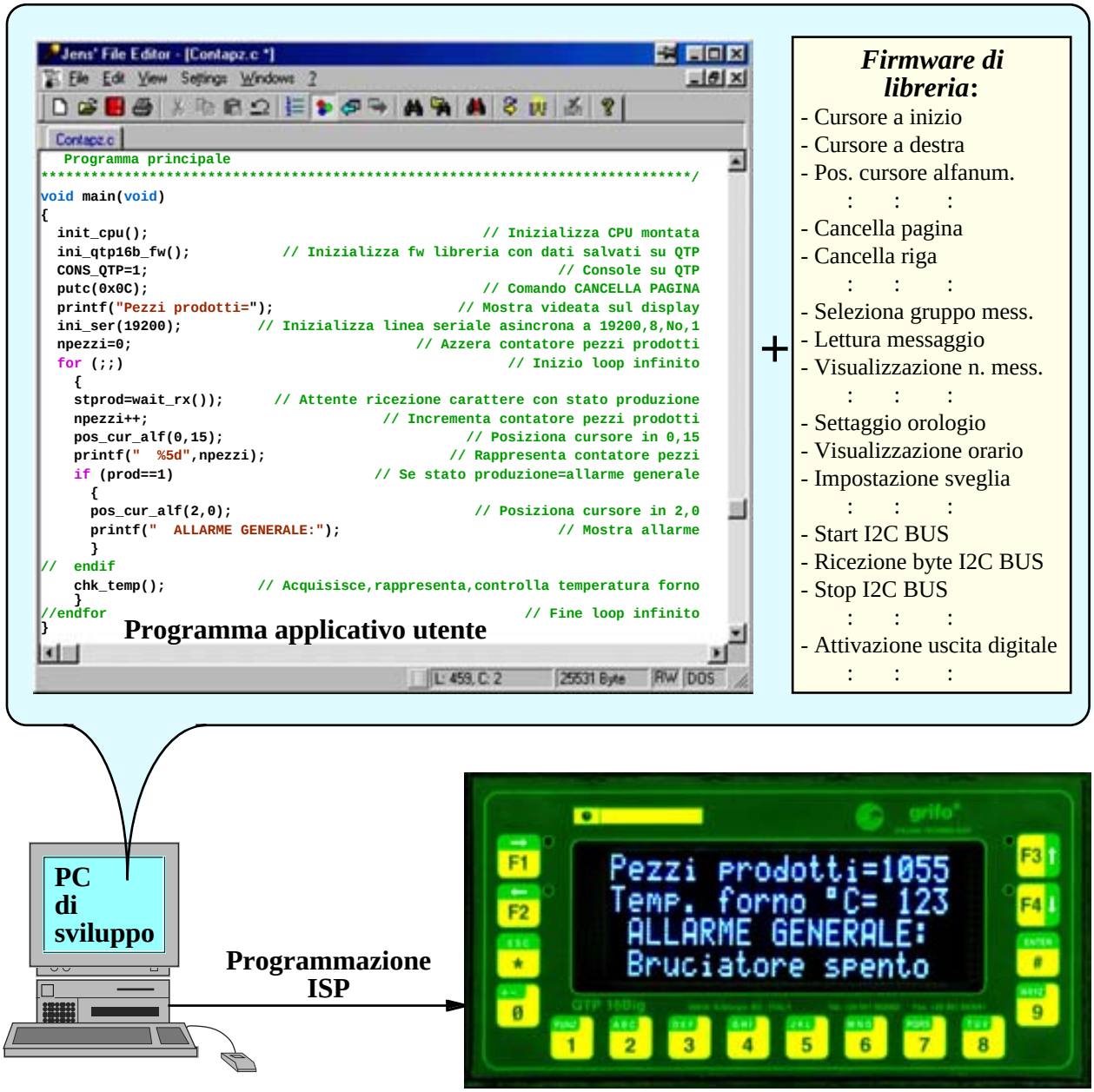


FIGURA 9: MODALITÀ SVILUPPO CON LIBRERIA

SETUP LOCALE

Nella **QTP 16Big.LIB** non é disponibile la modalità di setup locale descritta nel manuale utente della stessa scheda. Molte delle impostazioni definibili in questo setup non sono necessarie in quanto riguardano la comunicazione con l'unità di comando che nella libreria, non é più gestita. I rimanenti settaggi sono invece disponibili nella funzione INIZIALIZZA EEPROM descritta nei precedenti paragrafi.

MODALITÀ DI COMUNICAZIONE

A differenza del firmware di base della **QTP 16Big**, la libreria non prevede alcuna modalità di comunicazione verso unità di comando. Per questa ragione la linea seriale asincrona su CN5 non é usata dalla libreria, mentre la linea I2C BUS é usata dalla libreria solo in modalità master ma non in modalità slave.
In questo modo il programma applicativo può liberamente colloquiare con altri dispositivi sia in modalità punto punto che in rete, con qualsiasi protocollo fisico, logico ed elettrico. Tramite l'ambiente di sviluppo scelto sulla linea seriale si può inoltre effettuare il debug del programma applicativo, riducendo i tempi complessivi di preparazione.
Il protocollo fisico di comunicazione può essere definito dal programma applicativo utente tramite la programmazione di alcuni registri interni del microcontrollore In dettaglio si possono programmare 8,9 bit per carattere; parità pari, dispari o nessuna; 1 o 2 bit di stop; con baud rate standard e non standard, fino a 115200 Baud.
Il protocollo logico di comunicazione é altrettanto libero e sempre definito dal programma utente che può quindi implementare sia protocolli standard (MODBUS, XMODEM, ecc.) che propretari e comunicare quindi con tutti i sistemi presenti sul mercato, come stampanti, PC, modem, PLC, terminali, altri pannelli operatore, ecc.
Nel caso di comunicazioni RS 422 o RS 485 il programma applicativo utente si deve occupare anche di un apposito segnale denominato **DIR**, collegato al segnale **P2.1** del microcontrollore, come di seguito descritto:

Comunicazione RS 422 punto punto: sia il trasmettitore che il ricevitore possono essere sempre abilitati.

P2.1 = DIR = 0 -> trasmettitore attivo

Comunicazione RS 422 in rete: il ricevitore é sempre abilitato mentre il trasmettitore deve essere abilitato solo in fase di trasmissione.

P2.1 = DIR = 0 -> trasmettitore attivo
P2.1 = DIR = 1 -> trasmettitore disattivo

Comunicazione RS 485: il ricevitore é sempre abilitato ed il trasmettitore deve essere abilitato solo in fase di trasmissione, ottenendo la funzionalità di trasmissione o ricezione sulla linea half duplex.

P2.1 = DIR = 0 -> linea in trasmissione
P2.1 = DIR = 1 -> linea in ricezione

In questa modalità si riceve quanto trasmesso, in modo da fornire al sistema la possibilità di verificare autonomamente la riuscita della trasmissione; infatti in caso di conflitti sulla linea, quanto trasmesso non viene ricevuto correttamente e viceversa.

In fase di accensione, il segnale P2.1 = DIR è mantenuto a livello logico 1 di conseguenza in seguito ad una di queste fasi il driver RS 485 è in ricezione o il driver di trasmissione RS 422 è disattivo, in modo da eliminare eventuali conflitti sulla linea di comunicazione.

PROGRAMMI DEMO PER FIRMWARE DI LIBRERIA

In caso di primo acquisto tra il materiale ricevuto (dischi o CD) sono disponibili numerosi programmi dimostrativi che consentono di provare e valutare immediatamente il prodotto scelto. Tali programmi sono forniti in formato eseguibile e sorgente, sono corredati di tutti i file di contorno necessari (header, progetti, librerie, cinfigurazioni, ecc.) e sono disponibili per gli ambienti di sviluppo proposti da grifo®.

Come indicato nel paragrafo COME INIZIARE... i programmi con il nome *PRFWQ16B.** utilizzano tutti i comandi disponibili tramite una semplice iterazione con l'utente e consentono quindi di gestire il display in tutte le sue modalità di rappresentazione, la tastiera, i LEDs ed il buzzer, la/le EEPROM, i messaggi, le uscite digitali, l'orologio e la sua funzione di sveglia, alcune periferiche I2C BUS, ecc. Al fine di velocizzare l'uso degli esempi ed avere delle dimensioni compatibili con l'are codice lasciata libera dalla libreria, questi sono stati divisi in diversi programmi, ognuno dedicato ai principali gruppi di caratteristiche della libreria **QTP 16Big**. L'utente può esaminare il cartiglio di tali esempi e decidere autonomamente se provarli.
Tutti i programmi dimostrativi possono essere usati direttamente oppure modificati od utilizzati in parte, a seconda delle proprie esigenze, senza alcuna autorizzazione o costo aggiuntivo. In caso di particolari esigenze o combinazioni d'uso possono essere anche richiesti dei programmi specifici, oppure dei demo per diversi ambienti di sviluppo, previo accordo con la grifo®.

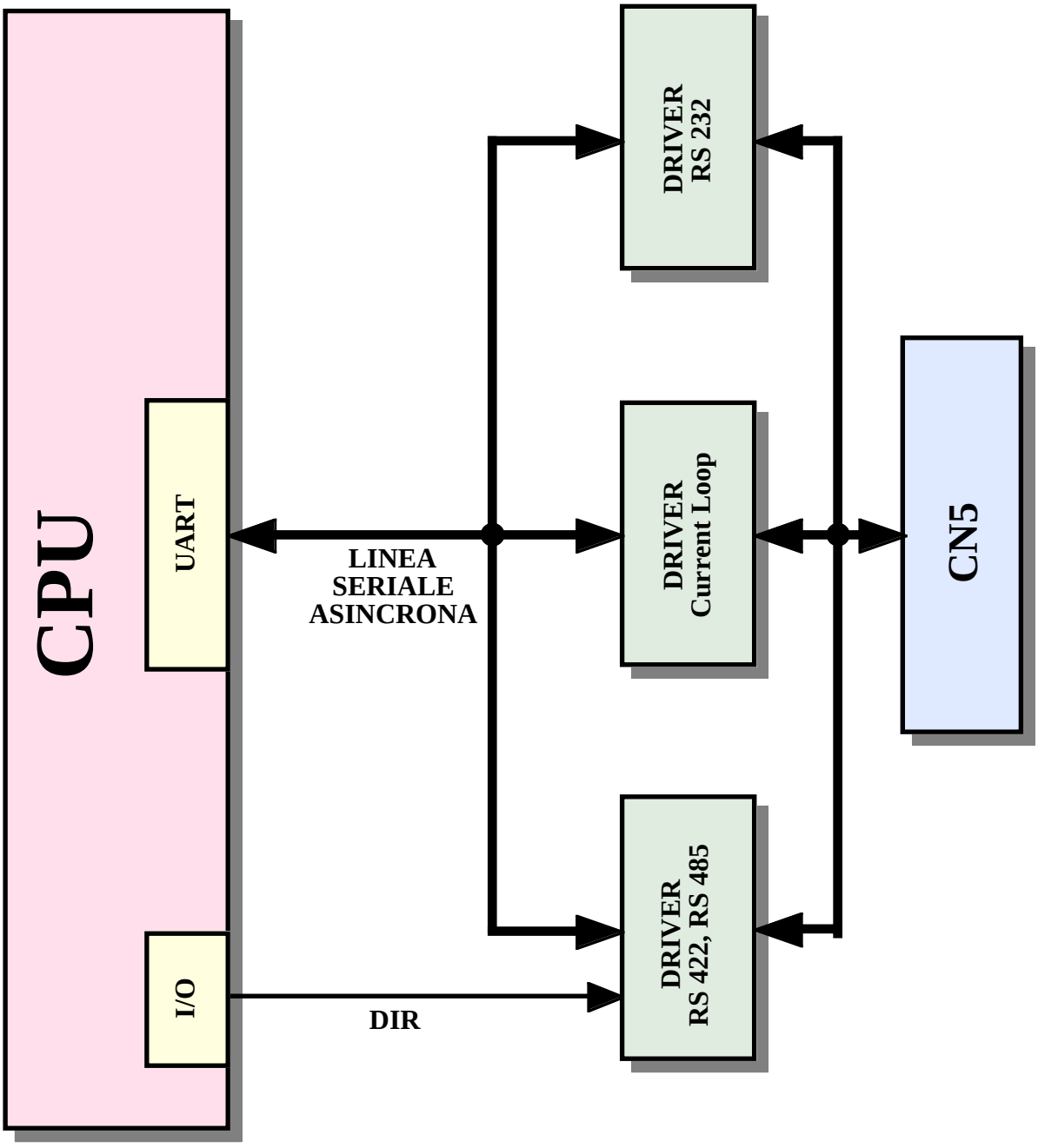


FIGURA 10: SCHEMA DI COMUNICAZIONE SERIALE

UTILIZZO LIBRERIA QTP 16BIG CON µC/51

In questo capitolo vengono riportate tutte le informazioni che riguardano l'integrazione e l'uso della libreria **QTP 16Big** con l'ambiente di sviluppo **µC/51**.
Il **µC/51** é un comodo pacchetto software, a basso costo, che tramite un completo IDE permette di utilizzare un editor, un compilatore ANSI C, un assemblatore, un linker e un remote debugger configurabile da utente, a livello sorgente. Sono inclusi i sorgenti delle librerie, interessanti programmi di utilità, documentazione d'uso, ed una ricca serie di esempi.
Si ricorda che questo capitolo riporta solo le informazioni sull'uso della libreria, non sul normale uso del **µC/51** e si consiglia quindi di esaminare la documentazione del pacchetto prima di esaminare i seguenti paragrafi.

INTEGRAZIONE LIBRERIA NEL µC/51

Facendo riferimento a quanto descritto nel paragrafo INTEGRAZIONE ED USO DELLA LIBRERIA, le tecniche usate per integrare la libreria nell'ambiente di sviluppo sono le seguenti:

- Sono state modificate le librerie del compilatore con le funzioni di console in modo da poterle ridirezionare sulle funzioni di console della libreria. In dettaglio per non perdere la console del **µC/51** associata alla linea seriale del microprocessore é stato previsto un flag, denominato *CONS_QTP*, che seleziona il dispositivo usato con la seguente corrispondenza:
  0 -> console associata alla linea seriale
  1 -> console associata alla libreria **QTP 16Big**
  Il flag *CONS_QTP* coincide con un bit di RAM interna allocato all'indirizzo **28H**.
  Il **µC/51** dispone di tre diverse librerie di console con altrettante modalità di gestione della seriale (P = polling, D = debugger, K = veloce) e tutte sono state modificate come descritto, aggiungendo la lettera Q come suffisso al loro nome originale (vedi paragrafo FILE FORNITI CON µC/51).

- Sono state riservate le aree di memoria RAM interna usate dalla libreria, provvedendo a definire tre appositi segmenti rispettivamente per l'area ad accesso diretto a bit, l'area ad accesso diretto a byte e l'area ad accesso indiretto a byte. Tali segmenti sono stati poi referenziati in una procedura di start up utente, che provvede ad informare il compilatore su quale memoria usare.

- E' stata ridirezionata la procedura di risposta all'interrupt del Timer 0.

- Sono state create due funzioni che chiamano le corrispondenti procedure di inizializzazione della libreria **QTP 16Big**. Tali funzioni sono state chiamate:
  ini_qtp16b_fw(); -> chiama la procedura **INIZIALIZZA FW**;
  ini_qtp16b_ee(fnrl2,exteesize); -> chiama la procedura **INIZIALIZZA EEPROM** con i due parametri passati, opportunamente salvati sui rispettivi registri ;

Tutte le tecniche descritte sono state radunate in un unico file, denominato *FWQTP16B.H*, che coincide con il file di dichiarazioni, necessarie all'uso della libreria **QTP 16Big**. Tale file deve essere sempre incluso nei programmi applicativi sviluppati dall'utente in quanto provvede a rendere immediatamente utilizzabile la stessa libreria.

FILES FORNITI CON µC/51

Segue una breve descrizione dei file necessari per sviluppare un programma applicativo utente con la libreria, usando il software di sviluppo **µC/51**. Tali file sono naturalmente forniti nelle cartelle del CD o disco ricevuto in abbinamento al primo acquisto del pannello operatore **QTP 16Big.LIB**.

\LIB\LARGE*.* -> File di libreria per console con modello di memoria *large* del **µC/51**. Questi file associano le istruzioni C di console alle funzioni del firmware di libreria quando il bit di RAM interna all'indirizzo 28H é settato e viceversa, alla linea seriale hw, quando il bit é resettato. Come nei file di libreria originali ci sono tre diverse modalità di gestione della console:

SER_IODQ.LIB = gestione in interrupt con debugger;
SER_IOPQ.LIB = gestione in polling con debugger;
SER_IOKQ.LIB = gestione in polling senza debugger;

Devono essere copiati manualmente sull'hard disk del PC di sviluppo, nella cartella *\UC51\LIB\LARGE* .

\LIB\SMALL*.* -> File di libreria per console con modello di memoria *small* del **µC/51**. Questi file associano le istruzioni C di console alle funzioni del firmware di libreria quando il bit di RAM interna all'indirizzo 28H é settato e viceversa, alla linea seriale hw, quando il bit é resettato. Come nei file di libreria originali ci sono tre diverse modalità di gestione della console:

SER_IODQ.LIB = gestione in interrupt con debugger;
SER_IOPQ.LIB = gestione in polling con debugger;
SER_IOKQ.LIB = gestione in polling senza debugger;

Devono essere copiati manualmente sull'hard disk del PC di sviluppo, nella cartella *\UC51\LIB\SMALL* .

QTP16Bxy.HEX -> File con codice eseguibile della libreria **QTP 16Big**, nella versione x.y, in formato HEX Intel.
Deve essere copiato manualmente nella cartella di lavoro del PC di sviluppo.

CANARY.H -> File di include con la dichiarazione dei registri delle periferiche interne del microcontrollore CANary usato sulla **QTP 16Big.LIB**.
Deve essere copiato manualmente nella cartella di lavoro del PC di sviluppo.

FWQ16B.H -> File di include con integrazione della libreria **QTP 16Big** (riserva aree di memoria, setta punti di entrata, ridireziona interrupts, ecc.).
Deve essere copiato manualmente nella cartella di lavoro del PC di sviluppo.

DL.BAT -> File di gestione download del codice generato sulla FLASH della **QTP 16Big.LIB** (tramite i file di comando descritti ai punti seguenti) e successivo lancio del programma di emulazione terminale HYPERTERMINAL.
Deve essere copiato manualmente nella cartella di lavoro del PC di sviluppo e deve essere modificato in relazione ai percorsi in cui sono salvati i programmi usati.

TERM19_COM1.HT -> File di configurazione per programma di emulazione terminale HYPERTERMINAL, che imposta il protocollo fisico di comunicazione sul PC di sviluppo, che così opera come console per il programma dimostrativo.
Deve essere copiato manualmente nella cartella di lavoro del PC di sviluppo.

PRFWQ16B.C -> File sorgente del primo programma dimostrativo che usa tutti i comandi standard della libreria **QTP 16Big**.
Deve essere copiato manualmente nella cartella di lavoro del PC di sviluppo.

PRFWQ16B.MAK -> File progetto per l'omonimo file sorgente.
Deve essere copiato manualmente nella cartella di lavoro del PC di sviluppo.

PRFWQ16B.HEX -> File eseguibile per l'omonimo file sorgente.
Deve essere copiato manualmente nella cartella di lavoro del PC di sviluppo.

PRFWQ16B.WSP -> File di predisposizione del JFE editor usato come completo IDE che consente all'utente di modificare, compilare e programmare l'omonimo programma dimostrativo.
Deve essere copiato manualmente nella cartella di lavoro del PC di sviluppo e deve essere modificato in relazione ai percorsi delle cartelle usate.

PRFWQ16B.CMD -> File di comando per la programmazione ISP della FLASH su **QTP 16Big.LIB**: effettua tutte le operazioni per salvare il programma demo e la libreria, usando la versione batch del FLIP.
Deve essere copiato manualmente nella cartella di lavoro del PC di sviluppo ed eventualmente modificato in relazione al microcontrollore usato..

PRFWQ16B_OPT.* -> Files del secondo programma dimostrativo che usa tutti i comandi per le opzioni ed accessori esterni, della libreria **QTP 16Big**.
Devono essere copiati manualmente nella cartella di lavoro del PC di sviluppo come descritto nei corrispondenti file *PRFWQ16B.**.

Con l'indicazione "cartella di lavoro del PC di sviluppo" s'intende la cartella generata sul PC di sviluppo in cui l'utente intende lavorare e quindi sviluppare il proprio programma applicativo. Riassumendo, in questa cartella devono essere copiati tutti file forniti ad eccezione di quelli di libreria. Si ricorda che nel tempo possono essere sviluppati altri programmi e/o files non riportati nella precedente lista: questi possono essere esaminati tramite il loro cartiglio.
Qualora i files descritti siano ricevuti su CD ROM, a seguito della loro copia sul PC di sviluppo, si devono verificare le proprietà dei files copiati ed assicurare che il campo *Sola lettura* sia disattivo.

COME INIZARE CON LIBRERIA QTP 16BIG E µC/51

In questo paragrafo vengono illustrate le operazioni da effettuare per iniziare ad usare la **QTP 16Big.LIB** in maniera rapida e lineare in abbinamento all'ambiente di sviluppo **µC/51**, senza dover affrontare e risolvere alcun problema iniziale.

A) *Collegamento seriale tra **QTP12.LIB** e PC di sviluppo:*

A1) Realizzare il collegamento seriale descritto nella figura 2 ovvero collegare i due segnali di comunicazione (TX RS232, RX RS232) e la massa di riferimento (GND) ad una porta di comunicazione COMx libera, del PC di sviluppo. Come si può notare tale cavo di collegamento é rovesciato e per praticità può essere ordinato alla grifo® specificando il codice CCR 9+9R.

A2) Avviare il programma di emulazione terminale HYPERTERMINAL sul PC di sviluppo ed impostare la comunicazione a:

Connetti	direttamente a COM x (quella usata al punto A1)
Bit per secondo	19200
Bit di Dati	8
Parità	Nessuna
Bit di Stop	1
Controllo di flusso	Nessuno

A3) Impostare la modalità RUN, ovvero non collegare il jumper J1.

A4) Fornire alimentazione su CN4 e verificare che: il buzzer si disattivi, che sul monitor del PC di sviluppo compaia la presentazione del programma demo e che a distanza di 5 secondi sul display compaia la stringa *Programma demo libreria QTP16Big*, con il cursore lampeggiante alla fine della stringa. Ogni **QTP 16Big.LIB**, in caso di primo acquisto, viene fornita con il rispettivo programma demo più la libreria già programmati nella FLASH e configurata per farlo partire all'accensione: se non vedete comparire le schermate iniziali descritte, riverificate le connessioni seriali e controllate che il jumper J1 sia aperto.

A5) Seguire le istruzioni del demo in modo da provare tutti i comandi della libreria e da verificarne gli effetti: l'utente deve interagire con il demo sia tramite la console seriale sul PC di sviluppo che tramite le risorse della stessa **QTP**.

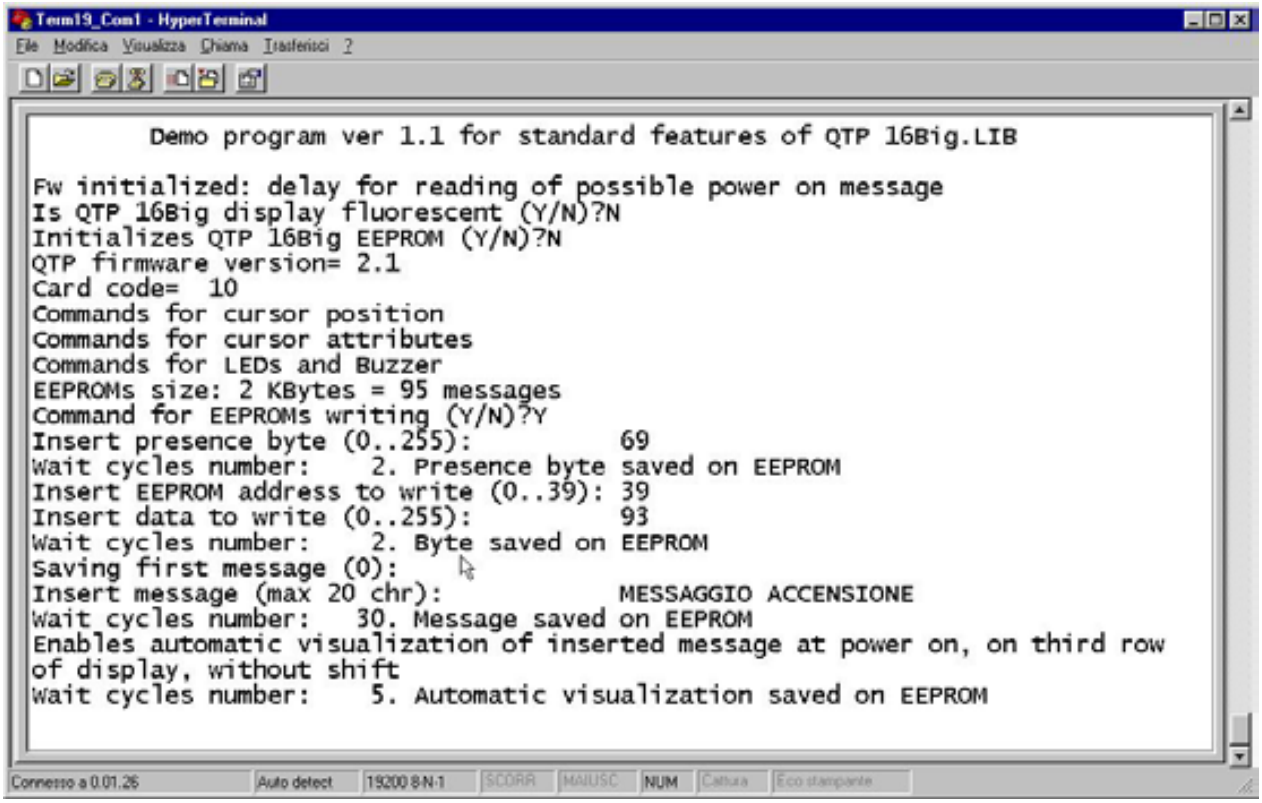

Term19_Com1 - HyperTerminal
File Modifica Visualizza Chiama Trasferisci ?
Demo program ver 1.1 for standard features of QTP 16Big.LIB
Fw initialized: delay for reading of possible power on message
Is QTP 16Big display fluorescent (Y/N)?N
Initializes QTP 16Big EEPROM (Y/N)?N
QTP firmware version= 2.1
Card code= 10
Commands for cursor position
Commands for cursor attributes
Commands for LEDs and Buzzer
EEPROMs size: 2 KBytes = 95 messages
Command for EEPROMs writing (Y/N)?Y
Insert presence byte (0..255): 69
Wait cycles number: 2. Presence byte saved on EEPROM
Insert EEPROM address to write (0..39): 39
Insert data to write (0..255): 93
Wait cycles number: 2. Byte saved on EEPROM
Saving first message (0):
Insert message (max 20 chr): MESSAGGIO ACCENSIONE
Wait cycles number: 30. Message saved on EEPROM
Enables automatic visualization of inserted message at power on, on third row of display, without shift
Wait cycles number: 5. Automatic visualization saved on EEPROM
Connesso a 0.01.26
Auto detect
19200 8-N-1
SCORR
MAIUSC
NUM
Cattura
Eco stampante

FIGURA 11: ESECUZIONE PROGRAMMA DEMO IN µC/51

A6) Terminata l'esecuzione del demo, togliere alimentazione alla **QTP**.

A7) Uscire dal programma HYPERTERMINAL sul PC di sviluppo.

B) *Riprogrammazione della FLASH con programma demo:*

B1) Sul disco **grifo®** ricevuto, localizzare e quindi installare sul disco rigido del PC di sviluppo il programma di utility **FLIP**. Questo gestisce la programmazione ISP della FLASH EPROM sulla **QTP 16Big.LIB**, come descritto nel precedente paragrafo PROGRAMMAZIONE FLASH EPROM.

B2) Installare l'ambiente di sviluppo **µC/51** sul disco rigido del PC di sviluppo e provvedere alla sua registrazione, in modo che non abbia i limiti della versione demo.

B3) Creare sull'hard disk del PC di sviluppo una cartella in cui si intende salvare tutto il lavoro svolto.

B4) Copiare i file di libreria descritti nel paragrafo FILES FORNITI CON µC/51, nelle cartelle con librerie create al punto B2, provvedendo ad eliminare l'eventuale proprietà di *Sola lettura*.

B5) Copiare tutti i rimanenti file descritti nel paragrafo FILES FORNITI CON µC/51, nella cartella di lavoro creata al punto B3, provvedendo ad eliminare l'eventuale proprietà di *Sola lettura*.

B6) Modificare il file *DL.BAT* con un semplice editor ASCII (ad esempio *BLOCCO NOTE* di Windows), andando ad impostare i giusti percorsi in cui sono salvati i programmi usati. Tali percorsi sono quelli in cui si trova il FLIP (installato al punto B1) ed il programma HYPERTERMINAL (installato con il sistema operativo Windows). Per facilità tali percorsi si possono ottenere dalla finestra proprietà degli stessi programmi e, come accade in tutti i file batch, devono essere inseriti con il formato MS-DOS standard. Quest'ultimo, in caso di nomi superiori agli 8 caratteri, prevede il mantenimento dei primi 6, l'aggiunta del carattere ~ e l'aggiunta di un numero progressivo che parte da 1 (ad esempio *\Programmi\Windows XP* diventa *\Progra~1\Window~1*).

B7) Modificare il file *PRFWQ16B.WSP* con un semplice editor ASCII (ad esempio *BLOCCO NOTE* di Windows), andando ad impostare il giusto percorso in cui si trova il **µC/51**. In dettaglio la modifica riguarda il percorso del compilatore a riga di comando *UMAKE.EXE* installato al punto B2, facilmente ottenibile anche dalla finestra proprietà dello stesso programma. Viste le dimensioni del file *PRFWQ16B.WSP*, si consiglia di cercare tutte le ricorrenze di *UMAKE.EXE* e quindi modificarne tutti i percorsi.

B8) Verificare che nel file *PRFWQ16B.CMD* sia selezionato il microcontrollore usato ed eventualmente modificarlo con un editor ASCII.

B9) Lanciare il *JFE editor* installato al punto B2, tramite il menù di avvio di Windows.

B10) Aprire il file di predisposizione *PRFWQ16B.WSP* usando l'apposito comando *File|Open Workspace* ed andandolo a selezionare sulla cartella di lavoro creata al ponto B3.

B11) Impostare la modalità DEBUG, ovvero collegare il jumper J1.

B12) Alimentare la **QTP 16Big.LIB** e verificare che il buzzer rimanga attivo.

B13) Premere *DL.BAT* sulla riga dei pulsanti del JFE. Con questa pressione si esegue la versione batch del FLIP che esegue il file di comandi *PRFWQ16B.CMD* che programma la FLASH.

B14) Attendere il completamento delle fasi della programmazione verificando che non avvenga alcun errore; a questo punto premere un tasto per continuare.

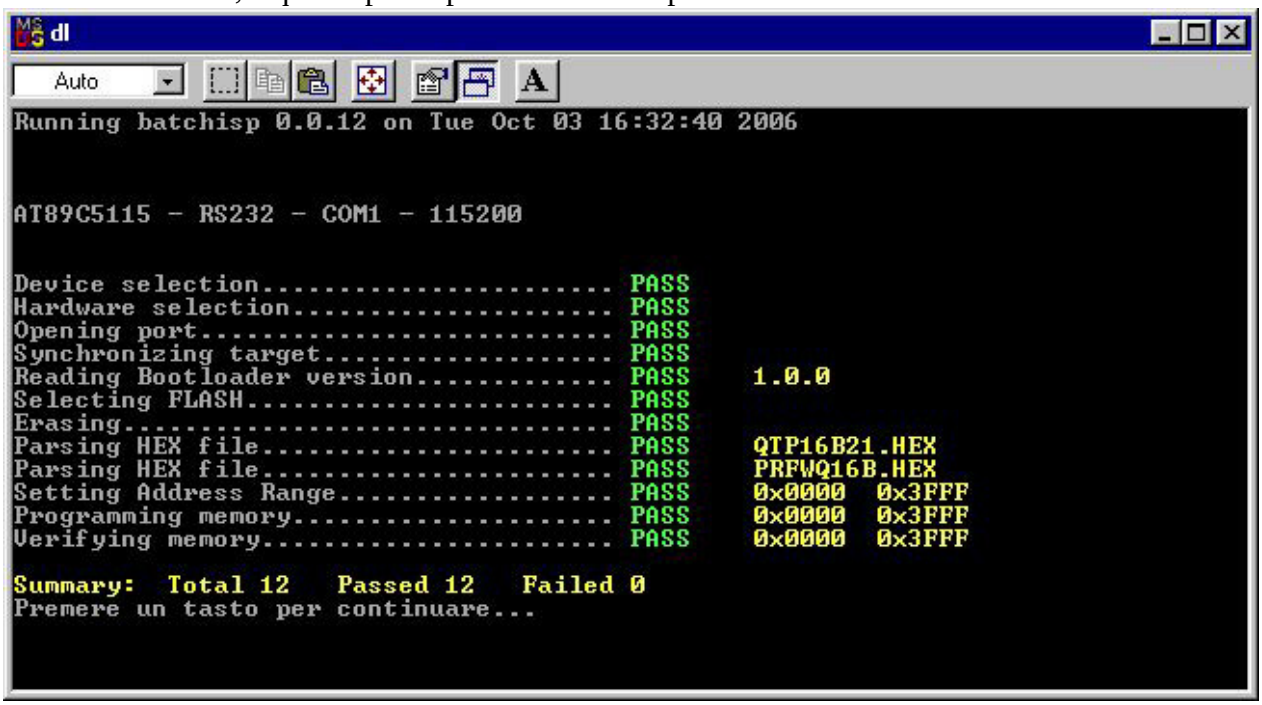

dl
Auto
Running batchisp 0.0.12 on Tue Oct 03 16:32:40 2006
AT89C5115 - RS232 - COM1 - 115200
Device selection..... PASS
Hardware selection..... PASS
Opening port..... PASS
Synchronizing target..... PASS
Reading Bootloader version..... PASS 1.0.0
Selecting FLASH..... PASS
Erasing..... PASS
Parsing HEX file..... PASS QTP16B21.HEX
Parsing HEX file..... PASS PRFWQ16B.HEX
Setting Address Range..... PASS 0x0000 0x3FFF
Programming memory..... PASS 0x0000 0x3FFF
Verifying memory..... PASS 0x0000 0x3FFF
Summary: Total 12 Passed 12 Failed 0
Premere un tasto per continuare...

FIGURA 12: PROGRAMMAZIONE FLASH EPROM CON µC/51

B15) Attendere la partenza del programma di emulazione terminale HYPERTERMINAL.

B16) Impostare la modalità RUN, ovvero non collegare il jumper J1.

B17) Spegnere e riaccendere la **QTP 16Big.LIB**.

B18) A questo punto il programma demo deve partire e funzionare come già accaduto al punto A.
B19) Uscire da HYPERTERMINAL in modo da tornare al JFE.

C) *Creazione del codice eseguibile del programma demo:*
C1) Ricompilare il sorgente del programma demo con una semplice pressione di *RE-MAKE* sulla riga dei pulsanti del JFE e verificare che non avvengano errori. In questo modo si deve ottenere il file *PRFWQ16B.HEX*, identico a quello presente sul disco **grifo®** e già usato nei punti B.

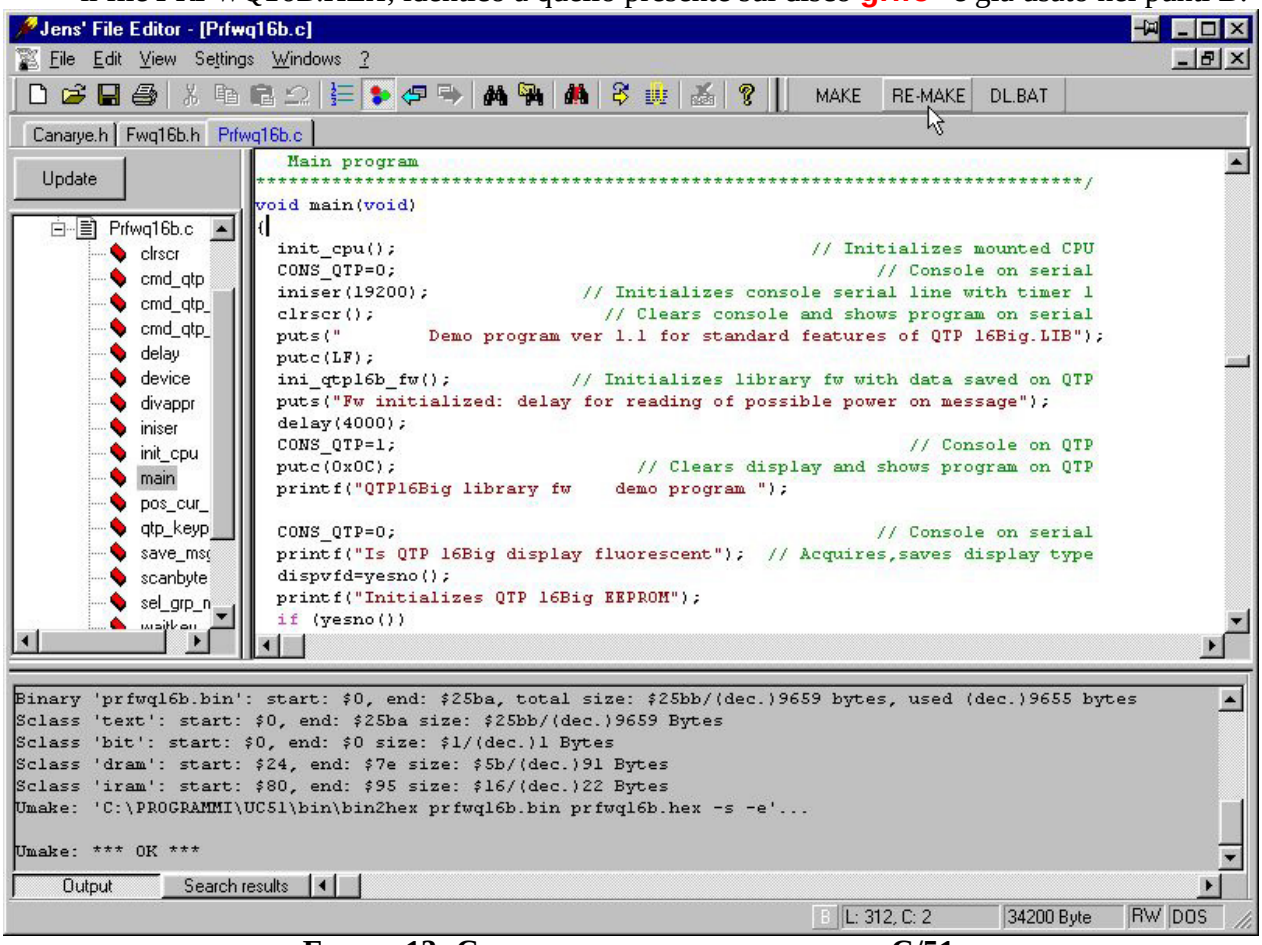


FIGURA 13: COMPILAZIONE PROGRAMMA CON µC/51

C2) Riprogrammare la FLASH con il codice del programma demo appena ricompilato, rieseguendo le operazioni dei punti B11÷B19.

Se durante l'esecuzione dei passi sopra elencati si presenta un problema od un'anomalia si consiglia all'utente di rileggere e ripetere i passi con attenzione e qualora il malfunzionamento persista, di contattare direttamente la **grifo®**.
In caso di esecuzione corretta di tutte le fasi sopra descritte l'utente ha realizzato e salvato il suo primo programma applicativo coincidente con il demo della **QTP 16Big.LIB**. A questo punto é possibile modificare il sorgente del/dei programmi demo in modo da soddisfare le richieste dell'applicazione da realizzare e ricompilarlo, riprogrammarlo e riverificarlo (passi da B11 a C2) in modo ciclico, fino a quando si ottiene la funzionalità desiderata. Raggiunto questo obiettivo si può eliminare il PC di sviluppo, ovvero:

D) *Preparazione definitiva dell'applicazione:*
D1) Impostare modalità RUN (jumper J1 non connesso) e scollegare PC di sviluppo se non richiesto dalla stessa applicazione.

UTILIZZO LIBRERIA QTP 16BIG CON BASCOM 8051

In questo capitolo vengono riportate tutte le informazioni che riguardano l'integrazione e l'uso della libreria **QTP 16Big** con l'ambiente di sviluppo **BASCOM 8051**.
Il **BASCOM 8051** é un comodo pacchetto software, a basso costo, che tramite un completo IDE permette di utilizzare un editor, un compilatore BASIC, un simulatore, un programmatore integrato od esterno ed un completo emulatore terminale. Sono inclusi numerose istruzioni espressamente dedicate all'automazione industriale, interessanti programmi di utilità, documentazione d'uso in linea, ed una ricca serie di esempi.
Si ricorda che questo capitolo riporta solo le informazioni sull'uso della libreria, non sul normale uso del **BASCOM 8051** e si consiglia quindi di esaminare la documentazione del pacchetto prima di esaminare i seguenti paragrafi.

INTEGRAZIONE LIBRERIA NEL BASCOM 8051

Facendo riferimento a quanto descritto nel paragrafo INTEGRAZIONE ED USO DELLA LIBRERIA, le tecniche usate per usare la libreria nell'ambiente di sviluppo sono le seguenti:

- Sono state ridirezionate le istruzioni di console del compilatore su due apposite procedure che si occupano di gestire l'ingresso e l'uscita di un carattere. In dettaglio il **BASCOM 8051** prevede già questa ridirezione con le direttive *$serialinput* e *$serialoutput* che specificano le relative procedure da eseguire. Per non perdere la console del **BASCOM 8051** associata alla linea seriale del microprocessore, nelle procedure di ridirezione console é stato previsto un flag, denominato *Cons_qtp*, che seleziona il dispositivo usato con la seguente corrispondenza:
  0 -> console associata alla linea seriale
  1 -> console associata alla libreria **QTP 16Big**
  Il flag *Cons_qtp* coincide con un bit di RAM interna dichiarato nel programma e quindi allocato direttamente dal compilatore.
  Le procedure di ridirezione sono scritte in assembly in modo da: salvare tutti i registri del microprocessore, chiamare direttamente i punti di entrata della libreria **QTP 16Big** e poter facilmente usare le variabili di ingresso ed uscita salvate nell'accumulatore.

- Visto che il **BASCOM 8051** non prevede la ridirezione dell'istruzione di stato della console (*Inkey*) é stata realizzata un'apposita procedura che effettua questa verifica, denominata *Consolesta*. Naturalmente la procedura agisce su entrambi i dispositivi fisici di console (linea seriale e libreria) a seconda del settaggio del solito flag *Cons_qtp* e restituisce sia lo stato di carattere disponibile che l'eventuale carattere pronto.

- Sono state riservate le aree di memoria RAM interna usate dalla libreria, tramite la direttiva *$iramstart* del compilatore, che viene impostata all'indirizzo di inizio dell'AREA PROGRAMMA APPLICATIVO UTENTE, riportata in figura 4. Inoltre nella finestra *Options|Compiler|Misc* del **BASCOM 8051** si setta il *Byte End(Hex)* ad un valore ottenuto dall'indirizzo di fine dell'AREA PROGRAMMA APPLICATIVO UTENTE, diminuito dell'area di stack; ad esempio prevedendo uno stack di 40 (=28H) bytes si dovrà settare tale valore a BF-28 = 97.

- E' stata ridirezionata la procedura di risposta all'interrupt del Timer 0, tramite le apposite direttive ed istruzioni ad alto livello offerte dal **BASCOM 8051** (*On Timer0.....*).

- Sono state create due funzioni che chiamano le corrispondenti procedure di inizializzazione della libreria **QTP 16Big**. Tali funzioni sono state chiamate:
  Ini_qtp16b_fw() -> chiama la procedura **INIZIALIZZA FW**;

Ini_qtp16b_ee(fnrl2,exteesize) -> chiama la procedura **INIZIALIZZA EEPROM** con i due parametri passati, opportunamente salvati sui rispettivi registri ;

Tutte le tecniche descritte devono essere presenti in un solo file che contiene anche il sorgente del programma applicativo sviluppato dall'utente. Fisicamente il tutto coincide con una serie di righe nel sorgente, opportunamente raggruppate ed identificate; tali righe devono essere sempre copiate nei programmi applicativi dell'utente in quanto provvedono a rendere immediatamente utilizzabile la stessa libreria.

FILES FORNITI CON BASCOM 8051

Segue una breve descrizione dei file necessari per sviluppare un programma applicativo utente con la libreria, usando il software di sviluppo **BASCOM 8051**. Tali file sono naturalmente forniti nelle cartelle del CD o disco ricevuto in abbinamento al primo acquisto del pannello operatore.

89C5115.DAT -> File di definizione caratteristiche dell'omonimo microprocessore usato sulla **QTP 16Big.LIB**.
Deve essere copiato manualmente nella cartella di intallazione del **BASCOM 8051** (*\MCS Electronics\BASCOM8051*) sull'hard disk del PC di sviluppo.

8951CC02.DAT -> File di definizione caratteristiche dell'omonimo microprocessore usato sulla **QTP 16Big.CAN.LIB**.
Deve essere copiato manualmente nella cartella di intallazione del **BASCOM 8051** (*\MCS Electronics\BASCOM8051*) sull'hard disk del PC di sviluppo.

QTP16Bxy.HEX -> File con codice eseguibile della libreria **QTP 16Big**, nella versione x.y, in formato HEX Intel.
Deve essere copiato manualmente nella cartella di lavoro del PC di sviluppo.

PRFWQ16B.BAS -> File sorgente del primo programma dimostrativo che usa tutti i comandi standard della libreria **QTP 16Big**.
Deve essere copiato manualmente nella cartella di lavoro del PC di sviluppo.

PRFWQ16B.HEX -> File eseguibile per l'omonimo file sorgente.
Deve essere copiato manualmente nella cartella di lavoro del PC di sviluppo.

PRFWQ16B.CMD -> File di comando per la programmazione ISP della FLASH su **QTP 16Big.LIB**: effettua tutte le operazioni per salvare il programma demo e la libreria, usando la versione batch del FLIP.
Deve essere copiato manualmente nella cartella di lavoro del PC di sviluppo ed eventualmente modificato in relazione al microcontrollore usato..

PRFWQ16B_OPT.* -> Files del secondo programma dimostrativo che usa tutti i comandi per le opzioni ed accessori esterni, della libreria **QTP 16Big**.
Devono essere copiati manualmente nella cartella di lavoro del PC di sviluppo come descritto nei corrispondenti file *PRFWQ16B.**.

Con l'indicazione "cartella di lavoro del PC di sviluppo" s'intende la cartella generata sul PC di sviluppo in cui l'utente intende lavorare e quindi sviluppare il proprio programma applicativo. Riassumendo, in questa cartella devono essere copiati tutti file forniti, ad eccezione di quelli di definizione con estensione .DAT. Si ricorda che nel tempo possono essere sviluppati altri programmi e/o files non riportati nella precedente lista: questi possono essere esaminati tramite il loro cartiglio. Qualora i files descritti siano ricevuti su CD ROM, a seguito della loro copia sul PC di sviluppo, si devono verificare le proprietà dei files copiati ed assicurare che il campo *Sola lettura* sia disattivo.

COME INIZARE CON LIBRERIA QTP 16BIG E BASCOM 8051

In questo paragrafo vengono illustrate le operazioni da effettuare per iniziare ad usare la **QTP 16Big.LIB** in maniera rapida e lineare in abbinamento all'ambiente di sviluppo **BASCOM 8051**, senza dover affrontare e risolvere alcun problema iniziale.

A) *Collegamento seriale tra **QTP12.LIB** e PC di sviluppo:*

A1) Realizzare il collegamento seriale descritto nella figura 2 ovvero collegare i due segnali di comunicazione (TX RS232, RX RS232) e la massa di riferimento (GND) ad una porta di comunicazione COMx libera, del PC di sviluppo. Come si può notare tale cavo di collegamento é rovesciato e per praticità può essere ordinato alla **grifo®** specificando il codice CCR 9+9R.

A2) Installare l'ambiente di sviluppo **BASCOM 8051** sul disco rigido del PC di sviluppo e provvedere alla sua registrazione ed eventuale aggiornamento all'ultima versione disponibile.

A3) Creare sull'hard disk del PC di sviluppo una cartella in cui si intende salvare tutto il lavoro svolto.

A4) Copiare i file di definizione descritti nel paragrafo FILES FORNITI CON BASCOM 8051, nella cartella in cui é stato installato l'ambiente di sviluppo al punto A2, provvedendo ad eliminare l'eventuale proprietà di *Sola lettura*.

A5) Copiare tutti i rimanenti file descritti nel paragrafo FILES FORNITI CON BASCOM 8051, nella cartella di lavoro creata al punto A3, provvedendo ad eliminare l'eventuale proprietà di *Sola lettura*.

A6) Lanciare il **BASCOM** installato al punto A2, tramite il menù di avvio di Windows.

A7) Aprire il file sorgente *PRFWQ16B.BAS* usando l'apposito comando *File|Open* ed andandolo a selezionare sulla cartella di lavoro creata al ponto A3.

A8) Aprire la finestra di configurazione dell'emulatore terminale del **BASCOM 8051**, tramite il comando *Option|Communication*, selezionare la linea seriale (*COM port*) collegata al punto A1, impostare il protocollo fisico di comunicazione descritto nella seguente figura e confermare con il pulsante *Ok*.

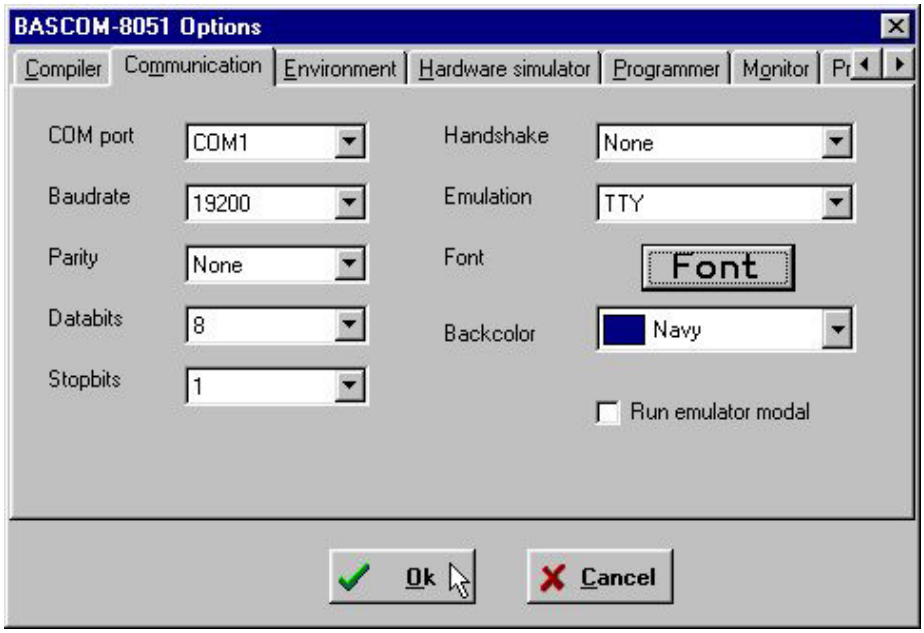


FIGURA 14: CONFIGURAZIONE SERIALE CON BASCOM 8051

A9) Aprire la finestra di emulazione terminale tramite il comando *Tools|Terminal emulator* e verificare che questa venga correttamente aperta.

A10) Impostare la modalità RUN, ovvero non collegare il jumper J1.

A11) Fornire alimentazione su CN4 e verificare che: il buzzer si disattivi, che nella finestra *BASCOM-8051 Terminal emulator* compaia la presentazione del programma demo e che a distanza di 5 secondi sul display compaia la stringa *Programma demo libreria QTP16Big*, con il cursore lampeggiante alla fine della stringa. Ogni **QTP 16Big.LIB**, in caso di primo acquisto, viene fornita con il rispettivo programma demo più la libreria già programmati nella FLASH e configurata per farlo partire all'accensione: se non vedete comparire le schermate iniziali descritte, riverificate le connessioni seriali e controllate che il jumper J1 sia aperto.

A12) Seguire le istruzioni del demo in modo da provare tutti i comandi della libreria e da verificarne gli effetti: l'utente deve interagire con il demo sia tramite la console seriale sul PC di sviluppo che tramite le risorse della stessa **QTP**.

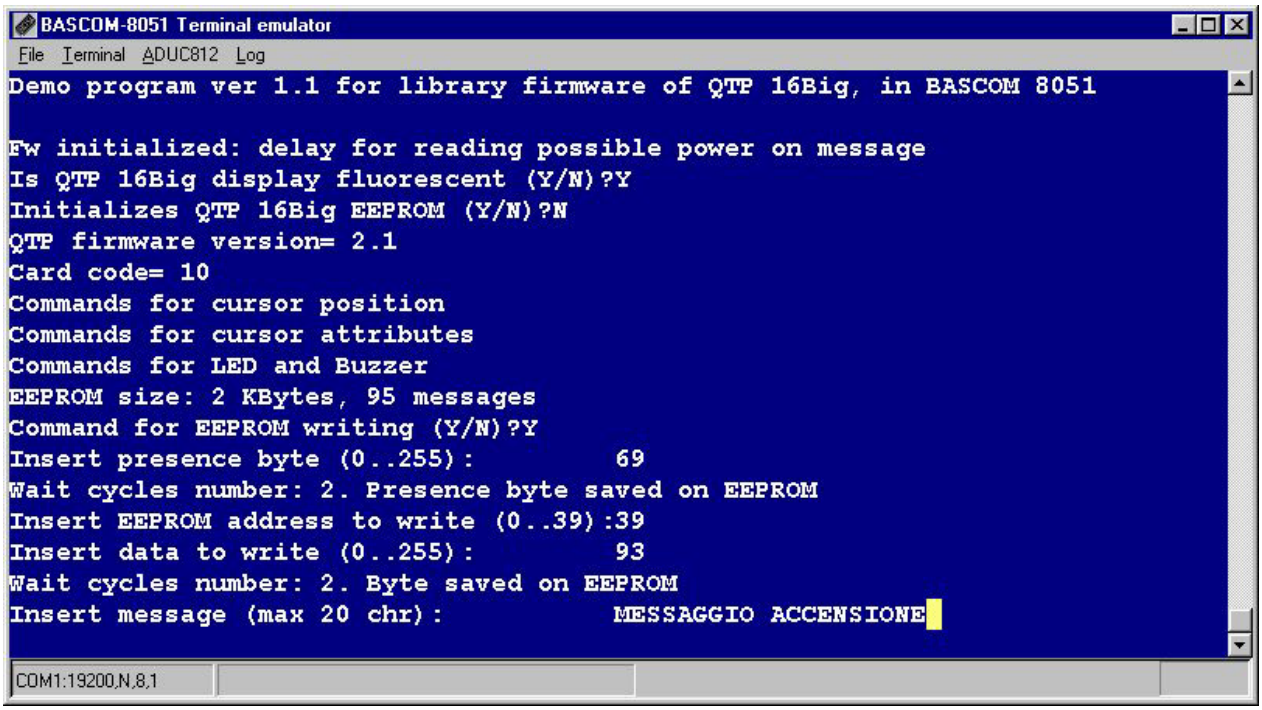

BASCOM-8051 Terminal emulator
File Terminal ADUC812 Log
Demo program ver 1.1 for library firmware of QTP 16Big, in BASCOM 8051
Fw initialized: delay for reading possible power on message
Is QTP 16Big display fluorescent (Y/N)?Y
Initializes QTP 16Big EEPROM (Y/N)?N
QTP firmware version= 2.1
Card code= 10
Commands for cursor position
Commands for cursor attributes
Commands for LED and Buzzer
EEPROM size: 2 KBytes, 95 messages
Command for EEPROM writing (Y/N)?Y
Insert presence byte (0..255): 69
Wait cycles number: 2. Presence byte saved on EEPROM
Insert EEPROM address to write (0..39):39
Insert data to write (0..255): 93
Wait cycles number: 2. Byte saved on EEPROM
Insert message (max 20 chr): MESSAGGIO ACCENSIONE
COM1:19200,N,8,1

FIGURA 15: ESECUZIONE PROGRAMMA DEMO IN BASCOM 8051

A13) Terminata l'esecuzione del demo, togliere alimentazione alla **QTP**.

A14) Chiudere la finestra di emulazione terminale tramite il comando *File|Exit*, o l'apposito pulsante *X* nell'angolo in alto a destra, della stessa finestra sul PC di sviluppo.

B) *Riprogrammazione della FLASH con programma demo:*

B1) Sul disco **grifo®** ricevuto, localizzare e quindi installare sul disco rigido del PC di sviluppo il programma di utility **FLIP**. Questo gestisce la programmazione ISP della FLASH EPROM sulla **QTP 16Big.LIB**, come descritto nel precedente paragrafo PROGRAMMAZIONE FLASH EPROM.

B2) Verificare che nel file *PRFWQ16B.CMD* copiato al punto A5 sia selezionato il microcontrollore usato ed eventualmente modificarlo con un editor ASCII.

B3) Aprire la finestra di configurazione del programmatore del **BASCOM 8051**, tramite il comando *Option|Programmer*, effettuare le impostazioni riportate nella seguente figura ed al termine confermare con il pulsante *Ok*.
Si ricorda che l'impostazione del campo *Program* coincide con la scelta del programma *BATCHISP.EXE* del FLIP, installato al punto B1.

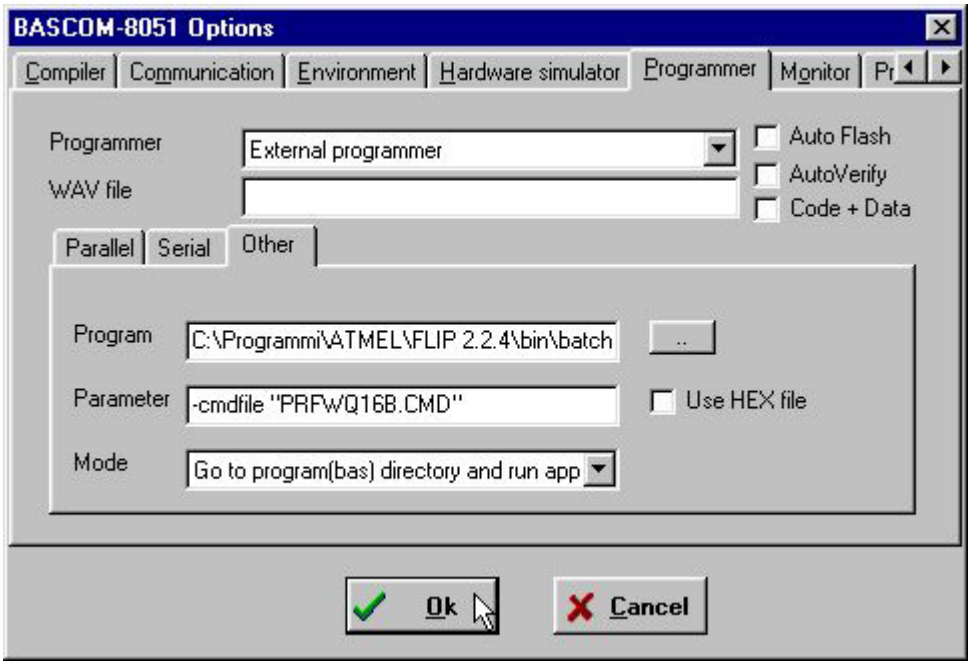


FIGURA 16: CONFIGURAZIONE PROGRAMMATORE CON BASCOM 8051

B4) Impostare la modalità DEBUG, ovvero collegare il jumper J1.

B5) Alimentare la **QTP 16Big.LIB** e verificare che il buzzer rimanga attivo.

B6) Premere il tasto rapido F4 oppure selezionare il comando *Program|Send to chip*. In questo modo si esegue la versione batch del FLIP, che esegue il file di comandi *PRFWQ16B.CMD*, che programma la FLASH

B7) Attendere il completamento delle fasi della programmazione verificando che non avvenga alcun errore; a questo punto premere un tasto per continuare.

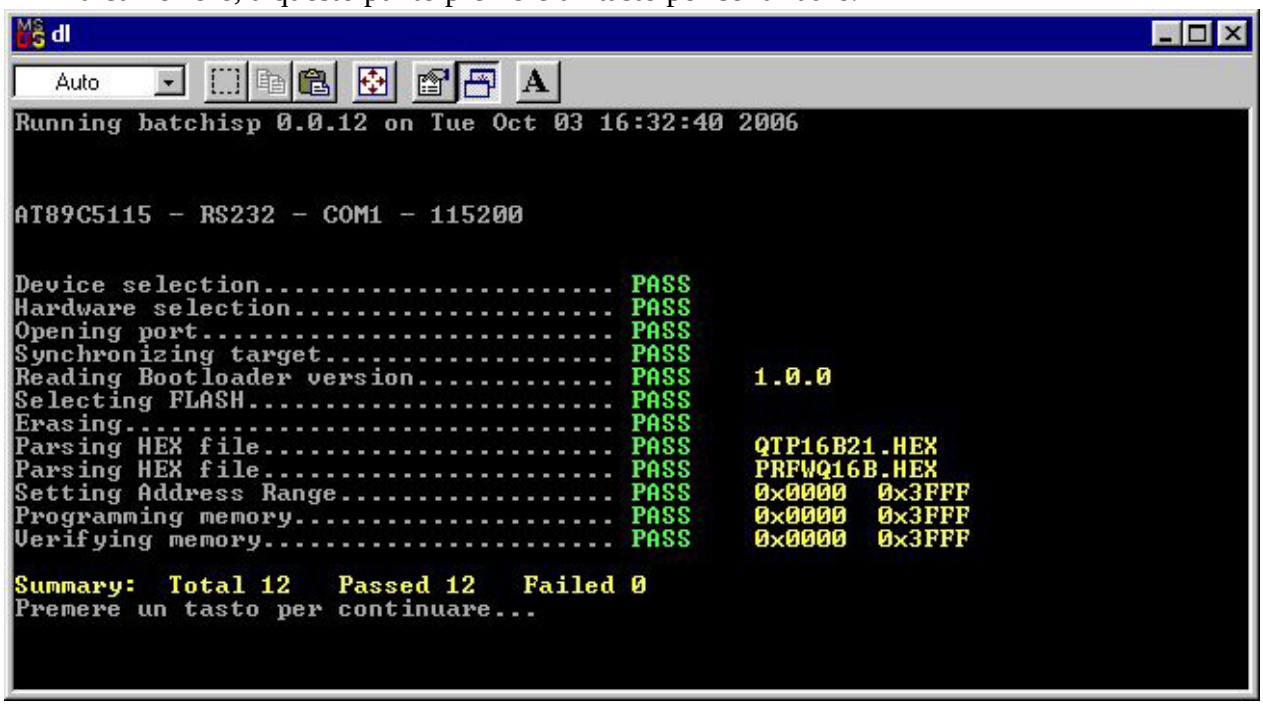

dl
Auto
Running batchisp 0.0.12 on Tue Oct 03 16:32:40 2006
AT89C5115 - RS232 - COM1 - 115200
Device selection..... PASS
Hardware selection..... PASS
Opening port..... PASS
Synchronizing target..... PASS
Reading Bootloader version..... PASS 1.0.0
Selecting FLASH..... PASS
Erasing..... PASS
Parsing HEX file..... PASS QTP16B21.HEX
Parsing HEX file..... PASS PRFWQ16B.HEX
Setting Address Range..... PASS 0x0000 0x3FFF
Programming memory..... PASS 0x0000 0x3FFF
Verifying memory..... PASS 0x0000 0x3FFF
Summary: Total 12 Passed 12 Failed 0
Premere un tasto per continuare...

FIGURA 17: PROGRAMMAZIONE FLASH EPROM CON BASCOM 8051

B8) Riaprire la finestra di emulazione terminale tramite il comando *Tools|Terminal emulator* od i tasti rapidi *Ctrl+T*.

B9) Impostare la modalità RUN, ovvero non collegare il jumper J1.
B10) Spegnere e riaccendere la **QTP 16Big.LIB**.
B11) A questo punto il programma demo deve partire e funzionare come già accaduto al punto A.
B12) Chiudere la finestra di emulazione terminale in modo da tornare all'IDE del **BASCOM 8051**.

C) *Creazione del codice eseguibile del programma demo:*
C1) Aprire la finestra di configurazione del compilatore **BASCOM 8051**, tramite il comando *Option|Compiler|Misc*, effettuare le impostazioni riportate nella seguente figura ed al termine confermare con il pulsante *Ok*.
Si ricorda che l'impostazione del campo *Register file* deve coincidere con il microprocessore usato ed é possibile solo se i file di definizione sono stati correttamente copiati, come descritto al punto A4.

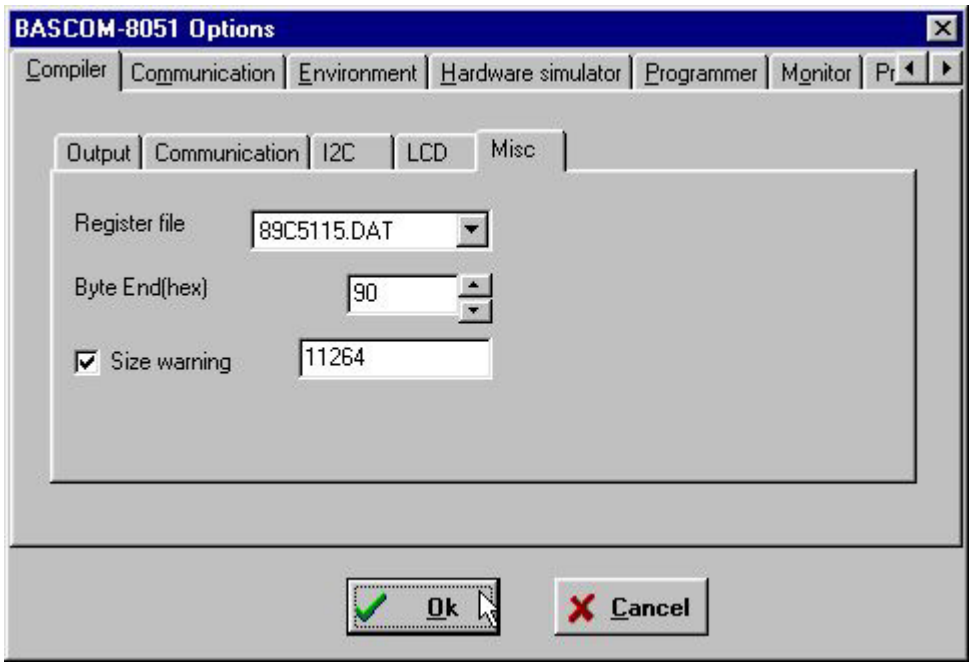


FIGURA 18: CONFIGURAZIONE COMPILATORE CON BASCOM 8051

C2) Ricompilare il sorgente del programma demo con una semplice pressione del tasto rapido *F7*, oppure selezionare il comando *Program|Compile*, e verificare che non avvengano errori. In questo modo si deve ottenere il file *PRFWQ16B.HEX*, identico a quello presente sul disco **grifo®** e già usato nei punti B.
La compilazione impiega un tempo che dipende dal PC usato, in ogni caso si deve attendere il completamento di entrambe le passate opportunamente segnalate da una finestra di stato che compare durante la compilazione, e verificare che al termine non siano riportati errori nella parte bassa dell'IDE. In altre parole al termine della compilazione si deve presentare una situazione simile a quella riportata nella seguente figura.

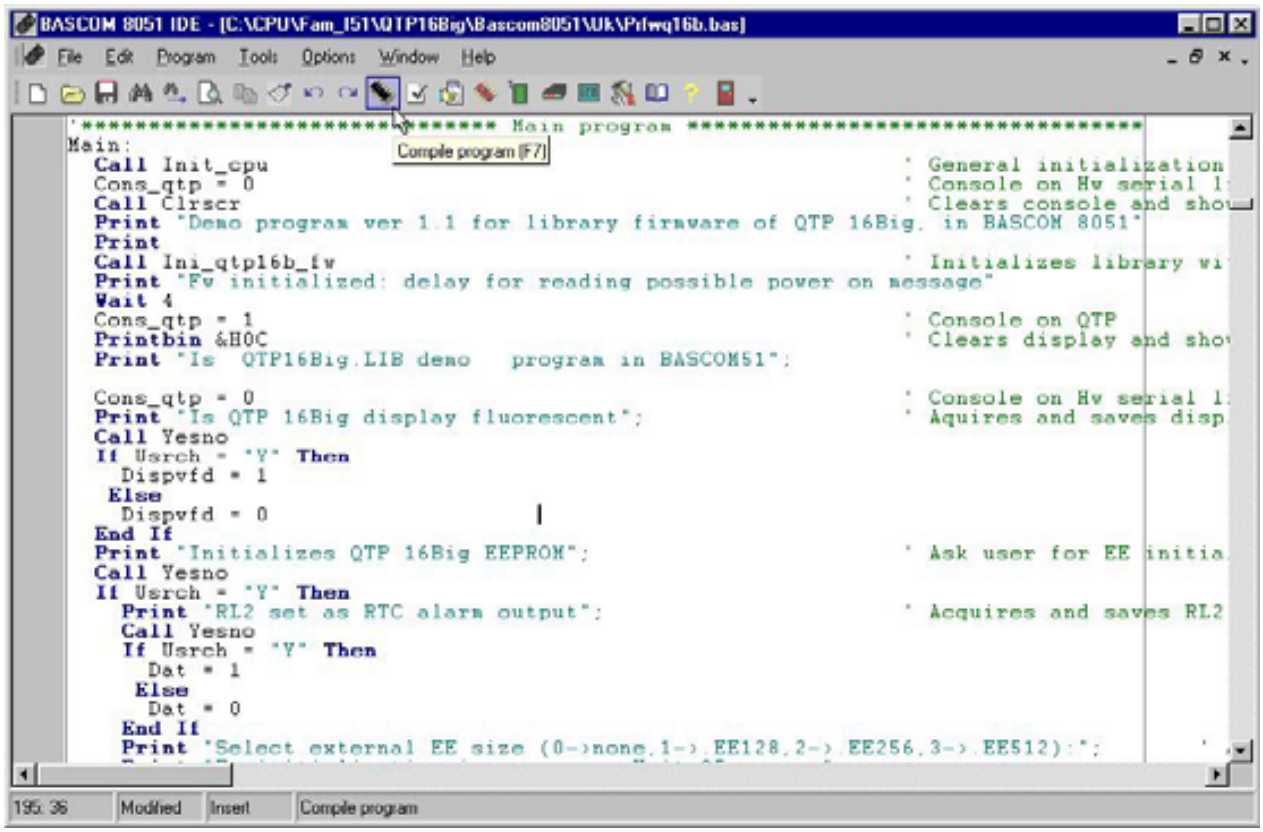

BASCOM 8051 IDE - [C:\CPU\Fam_I51\QTP16Big\Bascom8051\Uk\Prfwq16b.bas]
File Edit Program Tools Options Window Help
Compile program (F7)
'********** Main program **********
Main:
Call Init_cpu ' General initialization
Cons_qtp = 0 ' Console on Hv serial 1
Call Clrscr ' Clears console and sho
Print "Demo program ver 1.1 for library firmware of QTP 16Big. in BASCOM 8051"
Print
Call Ini_qtp16b_fw ' Initializes library wi
Print "Fw initialized: delay for reading possible power on message"
Wait 4
Cons_qtp = 1 ' Console on QTP
Printbin &H0C ' Clears display and sho
Print "Is QTP16Big.LIB demo program in BASCOM51";
Cons_qtp = 0 ' Console on Hv serial 1
Print "Is QTP 16Big display fluorescent"; ' Aquires and saves disp
Call Yesno
If Usrch = "Y" Then
Dispvfd = 1
Else
Dispvfd = 0
End If
Print "Initializes QTP 16Big EEPROM"; ' Ask user for EE initia
Call Yesno
If Usrch = "Y" Then
Print "RL2 set as RTC alarm output"; ' Acquires and saves RL2
Call Yesno
If Usrch = "Y" Then
Dat = 1
Else
Dat = 0
End If
Print "Select external EE size (0->none,1->.EE128,2->.EE256,3->.EE512):";
195: 36 Modified Insert Compile program

FIGURA 19: COMPILAZIONE PROGRAMMA CON BASCOM 8051

C3) Riprogrammare la FLASH con il codice del programma demo appena ricompilato, rieseguendo le operazioni dei punti B4÷B12.

Se durante l'esecuzione dei passi sopra elencati si presenta un problema od un'anomalia si consiglia all'utente di rileggere e ripetere i passi con attenzione e qualora il malfunzionamento persista, di contattare direttamente la **grifo®**.
In caso di esecuzione corretta di tutte le fasi sopra descritte l'utente ha realizzato e salvato il suo primo programma applicativo coincidente con il demo della **QTP 16Big.LIB**. A questo punto é possibile modificare il sorgente del/dei programmi demo in modo da soddisfare le richieste dell'applicazione da realizzare e ricompilarlo, riprogrammarlo e riverificarlo (passi da B4 a C3) in modo ciclico, fino a quando si ottiene la funzionalità desiderata. Raggiunto questo obiettivo si può eliminare il PC di sviluppo, ovvero:

D) *Preparazione definitiva dell'applicazione:*
D1) Impostare modalità RUN (jumper J1 non connesso) e scollegare PC di sviluppo se non richiesto dalla stessa applicazione.

Si ricorda che i passi di configurazione descritti ai punti A8, B3 e C1 devono essere normalmente eseguiti una sola volta, infatti il **BASCOM 8051** salva tutte le impostazioni del suo IDE all'uscita e le ricarica alla successiva esecuzione.

BIBLIOGRAFIA

E' riportato di seguito, un elenco di manuali e note tecniche, a cui l'utente può fare riferimento per avere maggiori chiarimenti, sui vari componenti montati a bordo della scheda **QTP 16Big.LIB**.

Manuale ATMEL:	*Microcontroller - AT89 series*
Manuale HEWLETT PACKARD:	*Optoelectronics Designer's Catalog*
Manuale MAXIM:	*New Releases Data Book - Volume IV*
Fogli tecnici NORITAKE ITRON:	*Dot VFD Modules*
Manuale PHILIPS:	*Application notes and development tools for 80C51 microcontrollers*
Manuale PHILIPS:	*80C51 - Based 8-Bit Microcontrollers*
Manuale PHILIPS:	*I^2C-bus compatible ICs*
Fogli tecnici CTC:	*LCD module specification*
Fogli tecnici S.E.:	*SI series - Switching power supply*
Manuale SGS-THOMSON:	*Small signal transistor - Data Book*
Manuale TEXAS INSTRUMENTS:	*RS-422 and RS-485 Interface Circuits*

Per reperire questi manuali fare riferimento alle case produttrici ed ai relativi distributori locali. In alternativa si possono ricercare le medesime informazioni o gli eventuali aggiornamenti ai siti internet delle case elencate.

APPENDICE A: DESCRIZIONE COMPONENTI DI BORDO

La **grifo®** fornisce un servizio di documentazione tecnica totalmente gratuito attraverso il proprio sito internet in cui possono essere scaricati le documentazioni tecniche complete dei componenti usati a bordo scheda. Si rimanda quindi l'utente a tali documenti scaricabili dalla pagina "Servizio Documentazion e Tecnica", di cui viene riportata solo la parte iniziale.

MICROCONTROLLORE T89C5115 O T89C51CC02

ATMEL WIRELESS & µC

T89C51CC02

8-bit MCU with CAN controller and Flash

1. Description

Part of the CANary™ family of microcontrollers dedicated to CAN network applications, the T89C51CC02 is a low pin count 8-bit Flash microcontroller.

While remaining fully compatible with the 80C51 it offers a superset of this standard microcontroller. In X2 mode a maximum external clock rate of 20 MHz reaches a 300 ns cycle time.

Besides the full CAN controller T89C51CC02 provides 16 Kbytes of Flash memory including In-system Programming (ISP), 2-Kbyte Boot Flash Memory, 2-Kbyte EEPROM and 512 bytes RAM.

Special attention is payed to the reduction of the electro-magnetic emission of T89C51CC02.

CANary™

2. Features

- 80C51 core architecture:
  - 256 bytes of on-chip RAM
  - 256 bytes of on-chip ERAM
  - 16 Kbytes of on-chip Flash memory
    Read/Write cycle : 10k
    Data Retention 10 years at 85°C
  - 2 Kbytes of on-chip Flash for Bootloader
  - 2 Kbytes of on-chip EEPROM
    Read/Write cycle : 100k
  - 14-source 4-level interrupt
  - Three 16-bit timer/counter
  - Full duplex UART compatible 80C51
  - maximum crystal frequency 40 MHz. In X2 mode, 20 MHz (CPU core, 40 MHz)
  - three or four ports: 16 or 20 digital I/O lines
  - two-channel 16-bit PCA with:
    - PWM (8-bit)
    - High-speed output
    - Timer and edge capture
  - Double Data Pointer
  - 21-bit watchdog timer (including 7 programmable bits)
- A 10-bit resolution analog to digital converter (ADC) with 8 multiplexed inputs
  - Separate power supply for analog
- Full CAN controller:
  - Fully compliant with CAN standard rev 2.0 A and 2.0 B
  - Optimized structure for communication management (via SFR)
  - 4 independent message objects:
    - Each message object programmable on transmission or reception
    - individual tag and mask filters up to 29-bit identifier/message object
    - 8-byte cyclic data register (FIFO)/message object
    - 16-bit status & control register/message object
    - 16-bit Time-Stamping register/message object
    - CAN specification 2.0 part A or 2.0 part B programmable message objects
    - Access to message object control and data register via SFR
    - Programmable reception buffer lenght up to 4 message objects
    - Priority management of reception of hits on several message objects at the same time (Basic CAN Feature)
    - Priority management for transmission
    - message object overrun interrupt
  - Supports
    - Time Triggered Communication.
    - Autobaud and Listening mode
    - Automatic reply mode programmable
  - 1 Mbit/s maximum transfer rate at 8MHz* Crystal frequency in X2 mode.
  - Readable error counters
  - Programmable link to on-chip Timer for Time Stamping and Network synchronization
  - Independent baud rate prescaler
  - Data, Remote, Error and overload frame handling
- Power saving modes:
  - Idle mode
  - Power down mode
- Power supply: 5V +/- 10% ,3V +/- 10%
- Temperature range: Industrial (-40° to +85°C)
- Packages: PLCC28, SOIC28, (TSSOP28, SOIC24)**

PROGRAMMER'S GUIDE AND INSTRUCTION SET

Memory Organization

Program Memory

The 80C51 has separate address spaces for program and data memory. The Program memory can be up to 64k bytes long. The lower 4k can reside on-chip. Figure 1 shows a map of the 80C51 program memory.

The 80C51 can address up to 64k bytes of data memory to the chip. The MOVX instruction is used to access the external data memory.

The 80C51 has 128 bytes of on-chip RAM, plus a number of Special Function Registers (SFRs). The lower 128 bytes of RAM can be accessed either by direct addressing (MOV data addr) or by indirect addressing (MOV @Ri). Figure 2 shows the Data Memory organization.

Direct and Indirect Address Area

The 128 bytes of RAM which can be accessed by both direct and indirect addressing can be divided into three segments as listed below and shown in Figure 3.

1. Register Banks 0-3: Locations 0 through 1FH (32 bytes). The device after reset defaults to register bank 0. To use the other register banks, the user must select them in software. Each register bank contains eight 1-byte registers 0 through 7. Reset initializes the stack pointer to location 07H, and it is incremented once to start from location 08H, which is the first register (R0) of the second register bank. Thus, in order to use more than one register bank, the SP should be initialized to a different location of the RAM where it is not used for data storage (i.e., the higher part of the RAM).
2. Bit Addressable Area: 16 bytes have been assigned for this segment, 20H-2FH. Each one of the 128 bits of this segment can be directly addressed (0-7FH). The bits can be referred to in two ways, both of which are acceptable by most assemblers. One way is to refer to their address (i.e., 0-7FH). The other way is with reference to bytes 20H to 2FH. Thus, bits 0-7 can also be referred to as bits 20.0-20.7, and bits 8-FH are the same as 21.0-21.7, and so on. Each of the 16 bytes in this segment can also be addressed as a byte.
3. Scratch Pad Area: 30H through 7FH are available to the user as data RAM. However, if the stack pointer has been initialized to this area, enough bytes should be left aside to prevent SP data destruction.

Figure 2 shows the different segments of the on-chip RAM.

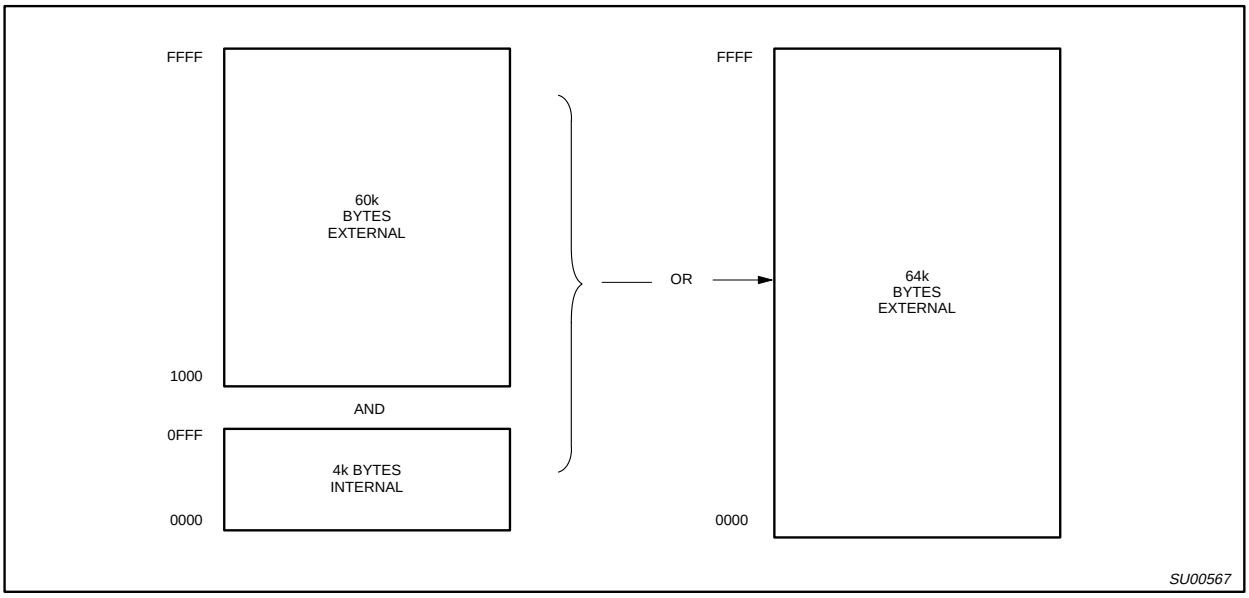


Figure 1. 80C51 Program Memory

EEPROM OPZIONALE

MICROCHIP

24AA512/24LC512/24FC512

512K I²C™ CMOS Serial EEPROM

Device Selection Table

Part Number	Vcc Range	Max. Clock Frequency	Temp. Ranges
24AA512	1.8-5.5V	400 kHz(1)	I
24LC512	2.5-5.5V	400 kHz	I, E
24FC512	2.5-5.5V	1 MHz	I

Note 1: 100 kHz for Vcc < 2.5V

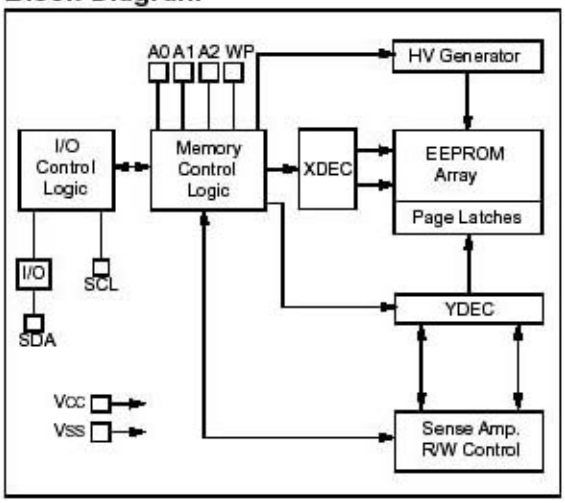
Features:

- Low-power CMOS technology:
  - Maximum write current 5 mA at 5.5V
  - Maximum read current 400 µA at 5.5V
  - Standby current 100 nA, typical at 5.5V
- 2-wire serial interface bus, I²C™ compatible
- Cascadable for up to eight devices
- Self-timed erase/write cycle
- 128-byte Page Write mode available
- 5 ms max. write cycle time
- Hardware write-protect for entire array
- Schmitt Trigger inputs for noise suppression
- 1,000,000 erase/write cycles
- Electrostatic discharge protection > 4000V
- Data retention > 200 years
- 8-pin PDIP, SOIC (208 mil), and DFN packages
- 14-lead TSSOP package
- Pb-free finishes available
- Temperature ranges:
  - Industrial (I): -40°C to +85°C
  - Automotive (E): -40°C to +125°C

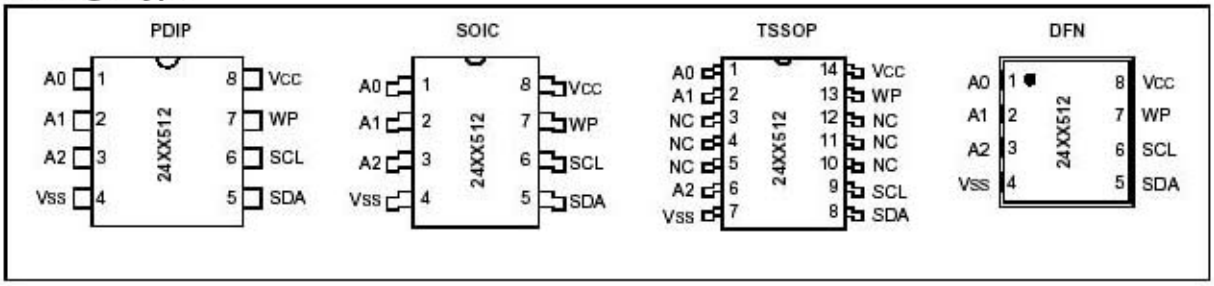
Description:

The Microchip Technology Inc. 24AA512/24LC512/24FC512 (24XX512*) is a 64K x 8 (512 Kbit) Serial Electrically Erasable PROM, capable of operation across a broad voltage range (1.8V to 5.5V). It has been developed for advanced, low-power applications such as personal communications and data acquisition. This device also has a page write capability of up to 128 bytes of data. This device is capable of both random and sequential reads up to the 512K boundary. Functional address lines allow up to eight devices on the same bus, for up to 4 Mbit address space. This device is available in the standard 8-pin plastic DIP, SOIC, DFN and 14-lead TSSOP packages.

Block Diagram



Package Type



* 24XX512 is used in this document as a generic part number for the 24AA512/24LC512/24FC512 devices.

© 2005 Microchip Technology Inc.

SRAM+RTC PCF8583

Clock/calendar with 240 × 8-bit RAM PCF8583

1 FEATURES

- I²C-bus interface operating supply voltage: 2.5 V to 6 V
- Clock operating supply voltage (0 to +70 °C): 1.0 V to 6.0 V
- 240 × 8-bit low-voltage RAM
- Data retention voltage: 1.0 V to 6 V
- Operating current (at f_{SCL} = 0 Hz): max. 50 μA
- Clock function with four year calendar
- Universal timer with alarm and overflow indication
- 24 or 12 hour format
- 32.768 kHz or 50 Hz time base
- Serial input/output bus (I²C)
- Automatic word address incrementing
- Programmable alarm, timer and interrupt function
- Slave address:
  - READ: A1 or A3
  - WRITE: A0 or A2.

2 GENERAL DESCRIPTION

The PCF8583 is a clock/calendar circuit based on a 2048-bit static CMOS RAM organized as 256 words by 8 bits. Addresses and data are transferred serially via the two-line bidirectional I²C-bus. The built-in word address register is incremented automatically after each written or read data byte. Address pin A0 is used for programming the hardware address, allowing the connection of two devices to the bus without additional hardware.

The built-in 32.768 kHz oscillator circuit and the first 8 bytes of the RAM are used for the clock/calendar and counter functions. The next 8 bytes may be programmed as alarm registers or used as free RAM space.
The remaining 240 bytes are free RAM locations.

3 QUICK REFERENCE DATA

SYMBOL	PARAMETER	CONDITION	MIN.	TYP.	MAX.	UNIT
V_{DD}	supply voltage operating mode	I²C-bus active	2.5	–	6.0	V
		I²C-bus inactive	1.0	–	6.0	V
I_{DD}	supply current operating mode	f_{SCL} = 100 kHz	–	–	200	μA
I_{DDO}	supply current clock mode	f_{SCL} = 0 Hz; V_{DD} = 5 V	–	10	50	μA
		f_{SCL} = 0 Hz; V_{DD} = 1 V	–	2	10	μA
T_{amb}	operating ambient temperature range		−40	–	+85	°C
T_{stg}	storage temperature range		−65	–	+150	°C

4 ORDERING INFORMATION

TYPE NUMBER	PACKAGE		
	NAME	DESCRIPTION	VERSION
PCF8583P	DIP8	plastic dual in-line package; 8 leads (300 mil)	SOT97-1
PCF8583T	SO8	plastic small outline package; 8 leads; body width 7.5 mm	SOT176-1

APPENDICE B: INDICE ANALITICO

Simboli

µC/51 **8**, **22**

A

Accensione **12**, **20**, **25**, **31**
Accessori **7**, **24**, **30**
Ambienti di sviluppo **8**, **10**, **19**, **22**, **28**
Area codice **11**
Assistenza **1**

B

BASCOM 8051 **8**, **28**
Baud rate **17**, **20**, **24**, **30**
Bibliografia **35**
Bit per carattere **20**, **24**, **30**
Boot loader **16**
Buffer ricezione **9**
Buffer trasmissione **9**

C

CAN **7**, **16**
Cariche elettrostatiche **1**
Cavo seriale **6**
CCR 9+9R **7**, **24**, **30**
Codice **11**, **14**
Collegamenti **6**
Come inizare con µC/51 **24**
Come inizare con BASCOM 8051 **30**
Compilazione programma **27**, **34**
Comunicazione seriale **6**, **20**, **24**, **30**
COMx **6**, **24**, **30**
Configurazione scheda **9**
Console **12**, **22**, **23**, **28**
Contenitore **1**
Controllo di flusso **24**
Creazione del codice **27**, **33**

D

DDS MICRO C 51 **8**
DEBUG **16**, **26**, **32**
Definizioni **29**
Descrizione **9**
DIR **20**

Direttive **1**
Diritti **2**

E

EEPROM **13**, **22**, **29**
EEPROM opzionale **13**, **A-3**
Emulatore terminale **7**, **24**, **30**
Entrata **11**
ETHERNET **7**
Evoluzione **3**

F

File definizione **29**
Files forniti **23**, **29**
FLASH EPROM **7**, **10**, **14**, **16**, **25**, **31**
FLIP **7**, **10**, **16**, **25**, **31**

G

Garanzia **1**
GET51 **7**

H

Handshake **24**, **30**
Header **20**, **23**
HI TECH C 51 **8**
HYPERTERMINAL **7**, **23**, **24**

I

Indicazioni generali **4**
Indirizzi **11**
Ingresso console **12**
Inizializzazione **13**, **22**, **28**
Integrazione libreria **10**, **22**, **28**
Interrupt **12**, **14**, **22**, **28**
Introduzione **1**
ISP **7**, **16**

J

Jumpers **9**, **16**, **24**, **31**

L

LADDER WORK **8**
Librerie **20**, **23**
Linea seriale **6**, **9**, **20**

M

Marchi registrati **2**
Materiale necessario **6**
Memorie di massa **6**
Microcontrollore **16**, **A-1**, **A-2**
Modalità **16**
Monitor **6**
Mouse **6**

N

NO OUT2 **13**

O

Opzioni **24**, **29**

P

Parità **20**, **24**, **30**
PC di sviluppo **6**
Personal Computer **6**
Preparazione definitiva applicazione **27**, **34**
Progetti **20**, **23**
Programma applicativo **19**, **20**
Programma emulazione terminale **7**, **24**, **30**
Programmazione FLASH **26**, **32**
Programmi demo **20**, **23**, **25**, **29**, **31**
Protezioni **1**
Protocollo fisico **20**, **24**, **30**
Protocollo logico **20**
Punti di entrata **11**, **12**

R

Rallentamenti **9**, **14**
RAM **14**, **15**, **22**, **28**
Rete **20**
RL2 **13**
Risorse usate **14**
Ritardi **9**
RS 232 **7**, **9**
RS 422 **20**
RS 485 **20**
RTC **40**
RUN **16**, **24**, **26**, **31**, **32**

S

Seriale **20**, **24**, **30**

Setup locale **19**
Sicurezza **1**
Sistema operativo **6**
Software **7**
SRAM+RTC **A-4**
Stack **14**, **28**
Stato console **12**, **28**
Stop bit **20**, **24**, **30**
SYS51CW **8**
SYS51PW **8**

T

T89C5115 **16**, **18**, **A-1**
T89C51CC02 **16**, **A-1**
Temporizzazioni **14**
Timeout Error **17**
Timer **14**, **22**, **28**

U

USB **7**
Uscita console **12**
Uscita digitale **13**

V

Versione **3**
Vettore **14**